SEMANTIC LOCALIZATION AND MAPPING IN ROBOT VISION

Roy C Anati

A DISSERTATION

in

Computer and Information Science

Presented to the Faculties of the University of Pennsylvania in Partial

Fulfillment of the Requirements for the Degree of Doctor of Philosophy

2016

Konstantinos Daniilidis, Professor
Computer and Information Science
Supervisor of Dissertation

Lyle Ungar, Professor
Computer and Information Science
Graduate Group Chairperson

Dissertation Committee

Camillo J. Taylor, Professor
Computer and Information Science
University of Pennsylvania

Daniel D. Lee, Professor
Computer and Information Science
University of Pennsylvania

Jianbo Shi, Professor
Computer and Information Science
University of Pennsylvania

Davide Scaramuzza, Assistant Professor
Department of Informatics
University of Zurich

SEMANTIC LOCALIZATION AND MAPPING IN ROBOT VISION

To my husband John, you were my anchor for this entire journey.

I couldn't have done it without you.

Acknowledgments

I want to thank my husband John for his complete and tireless support throughout the years. Thanks to my family, who helped me get started in the United States, and have been understanding of my infrequent visits. My advisor, Kostas Daniliildis, without whom this wouldn't have happened. I also want to thank my dissertation committee for their patience and feedback. Special thanks to Prof. Ben Taskar, who was instrumental to me publishing my first paper. I also want to thank my collegues in the GRASP lab for their help and advice. Finally a big thank you to my friends Yotam and Rachael, my friends in the Society for Creative Anachronisms (SCA) and my friends in the CIS department for believing in me.

ABSTRACT

SEMANTIC LOCALIZATION AND MAPPING IN ROBOT VISION

Roy C Anati

Konstantinos Daniilidis

Integration of human semantics plays an increasing role in robotics tasks such as mapping, localization and detection. Increased use of semantics serves multiple purposes, including giving computers the ability to process and present data containing human meaningful concepts, allowing computers to employ human reasoning to accomplish tasks.

This dissertation presents three solutions which incorporate semantics onto visual data in order to address these problems. First, on the problem of constructing topological maps from sequence of images. The proposed solution includes a novel image similarity score which uses dynamic programming to match images using both appearance and relative positions of local features simultaneously. An MRF is constructed to model the probability of loop-closures and a locally optimal labeling is found using Loopy-BP. The recovered loop closures are then used to generate a topological map. Results are presented on four urban sequences and one indoor sequence.

The second system uses video and annotated maps to solve localization. Data association is achieved through detection of object classes, annotated in prior maps, rather than through detection of visual features. To avoid the caveats of object recognition, a new representation of query images is introduced consisting of a vector of detection scores for each object class. Using soft object detections, hypotheses about pose are refined through particle filtering. Experiments include both small office spaces, and a large open urban rail station with semantically ambiguous places. This approach showcases a representation that is both robust and can exploit the plethora of existing prior maps for GPS-denied environments while avoiding the data association problems encountered when matching point clouds or visual features.

Finally, a purely vision-based approach for constructing semantic maps given camera pose and simple object exemplar images. Object response heatmaps are combined with

known pose to back-project detection information onto the world. These update the world model, integrating information over time as the camera moves. The approach avoids making hard decisions on object recognition, and aggregates evidence about objects in the world coordinate system.

These solutions simultaneously showcase the contribution of semantics in robotics and provide state of the art solutions to these fundamental problems.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Integration of semantics plays an increasing role in many robotics tasks such as mapping, localization and detection. Both localization and mapping are moving beyond metric representations to semantic descriptions with labeled components [Murillo and Kosecka, 2009; Pronobis and Jensfelt, 2012; Singh and Kosecka, 2012]. While recent detection approaches rely on more advanced features and inputs containing semantic information [Li et al., 2010b; Torresani et al., 2010; Yao et al., 2012]. This transition from low-level signals to meaningful symbols is based on a robot's ability to successfully recognize objects, scenes, and represent their relations. This permits computers to process input containing human meaningful concepts such as annotated maps [Apostolopoulos et al., 2012]. It also allows machines to produce output with advanced human concepts such as street intersections [Singh and Kosecka, 2012], or rooms [Pronobis and Jensfelt, 2012]. Semantics also enable computers to employ human reasoning to accomplish various tasks such as object based mapping [Galindo et al., 2005; Vasudevan et al., 2007] or object detection [Russakovsky and Ng, 2010; Torresani et al., 2010]. In this thesis, three solutions are presented which incorporate semantics to visual data in order to address these important problems in robotics.

1.1 Problem Statement

The application of semantics in robotics is an extremely broad topic covering many challenges. In this dissertation, the focus is placed firmly on expanding robot understanding of human environments. To that end, three problems are explored covering mapping, localization, and detection. In topological mapping, the goal is to incorporate spatial information to create a graph of the environment. For semantic localization, human concepts are used directly to localize a robot. Finally, in object detection, semantic constructs are used to detect and localize objects in a scene.

1.1.1 Topological Mapping

Under the umbrella definition of semantic mapping, topological mapping refers to the specific sub-problem of creating spatially meaningful maps without using an all metric representation. More formally a topological map is defined as:

Definition 1.1 (Topological Map)**.** A topological map T is a graph $T = (V, E)$, where V is a set of *distinct locations* and E edges describing connectivity between locations. Two locations $i, j \in V$, are connected by an edge (i, j) if and only if it is possible to reach location j from location i without physically passing through any other location $k \in V$.

Note the use of the phrase *distinct locations* in the definition. It is this aspect of the definition which allows us to categorize topological mapping as a semantic mapping problem. Although it is possible to define *distinct locations* in a metric fashion as Pronobis [2011] do, more semantic definitions exist. For Fraundorfer et al. [2007] and Cummins and Newman [2008a,b] locations are observed images. In the context of indoor mapping, Tomatis et al. [2003] define distinct locations as different rooms or corridors. On the other hand, Singh and Kosecka [2012] use street intersections for outdoor settings. The definition used in chapter 3 however is based on a combination of visual similarity and spatio-temporal reasoning.

1.1.2 Semantic Localization

The complement of mapping an environment is the ability to navigate it. The primary purpose of creating a map is to enable localization and facilitate navigation. More formally:

Definition 1.2 (Localization)**.** Localization combines prior information, such as a map or motion model, with sensor input, visual or otherwise, to recover the location of an agent.

By extension semantic localization is:

Definition 1.3 (Semantic Localization)**.** *Semantic* localization extends localization by employing prior information containing semantics, such as semantically annotated maps, or process sensor input to generate semantically meaningful results which are then used to perform localization.

To truly incorporate semantics in localization, it is not sufficient to produce a semantic labeling for a location. Rather, a semantic localization system uses semantic information to perform the localization itself. The majority of existing localization approaches employ metric sensors such as GPS, LIDAR [Bosse et al., 2003; Tomatis et al., 2003], or strictly visual appearance information [Fraundorfer et al., 2007; Murillo and Kosecka, 2009; Valgren and Lilienthal, 2007] or a combination of the above [Kumar et al., 2008; Pronobis and Jensfelt, 2012] but do not employ semantics. Scene *classification* such as Pronobis et al. [2010a,c]; Rottmann et al. [2005] can provide semantic labeling to a scene, such as "kitchen" or "office" but is unable to distinguish between two separate instances of the same class or incorporate spatial relationships between scenes. They can however be used to enhance metric maps to improve localization accuracy. In chapter 4 a semantic approach is presented which uses objects to successfully localize in an environment.

1.1.3 Object Detection, Recognition and Localization

Modeling an environment on a large scale is relevant for mapping and navigation, but does not provide understanding on a small scale. The ability to detect, recognize and

localize objects in a scene not only forms the basis for manipulation and interaction but also serves to accomplish not only scene classification [Pronobis et al., 2010a; Rottmann et al., 2005] but localization as well [Apostolopoulos et al., 2012]. Before we proceed, it is critical to emphasize the difference between image-based object localization approaches, and scene-based object localization approaches. The former seeks to localize an object in an image or video (whether by bounding box or segmentation), and does not build a model of the environment. While the latter, on the other hand, localizes an object within a scene and perforce generates an annotated map. It is the scene-based approach which is of greater interest in this dissertation, as that is more naturally adapted to applications such as mapping, manipulation and navigation. To be more specific:

Definition 1.4 (Object Localization in a Scene)**.** Object localization in a *scene* creates an annotated map of an environment with the location and *classification* of objects within it. A un-annotated map of the environment can serve as input or be generated from scratch.

Object *classification* is an integral part of the definition, and requires both *detection* and *recognition* of objects. Within scene-based, distinctions can be made across several dimensions:

- Sensor modalities: LIDAR [Anguelov et al., 2002] vs. vision [Bao et al., 2012; Lopez et al., 2008] vs. RGB-D [Lai et al., 2012].
- Planar objects [Castle and Murray, 2009; Lopez et al., 2008] vs. solid objects [Anguelov et al., 2002].
- Object *detection*/*recognition* stage: Image [Bao et al., 2012; Lopez et al., 2008] vs. 3D [Lai et al., 2012].
- 3D object pose [Lai et al., 2012; Lopez et al., 2008] vs. 2D object pose [Anguelov et al., 2002; Bao et al., 2012].

The work outlined in chapter 5 is preliminary work describing a vision only approach

to localizing objects in a scene. Objects are detected and recognized in 2D images, but localized in 3D to create a 3D annotated map.

1.2 Motivation

Although many mapping, localization and detection solutions exist the majority fall into one or more of the following categories:

- Not pure vision, which include other sensor modalities such as GPS, LIDAR, depth sensors (e.g. Kinect).
- 3D metric approaches that model the environment in an absolute metric scale.
- Scene *classification* instead of true *localization*. These approaches identify the *class* of room or scene, but contain no spatial interpretation. Often these rely on environments where a single instance exists for each scene class.
- Limited, or no semantics. Referring to *semantic* as defined earlier, many "semantic" approaches either contain no human semantics, or do so tangentially, where the internal representation and reasoning contains not human meaningful components.

A survey of works which fall into these categories is covered in chapter 2. These properties do not prevent existing solutions from meeting success or having relevance, but do pose significant limitations explained below. This dissertation proposes solutions to mapping and localization which avoid these.

The desire to devise purely vision based solutions stems from two causes. First, the inability of other sensors to function under various conditions. For example, GPS in indoor environments, infra-red sensors (such as the Kinect) in outdoor environments. The prevalence of cameras in mobile electronics such as mobile phone and tablet computers, in addition to the low cost of consumer level digital cameras, is the second reason. A massive and ubiquitous presence of visual sensors makes large-scale deployment a possibility.

Metric approaches on the other hand emphasize spatial precision. This emphasis not only entails increased cost in both processing and storage but is also less tolerant of errors. For applications involving robot grasping, or 3D modeling such precision is desired and required, but can be unnecessary in cases of planning, localization and human interaction. Although consumer GPS navigation provides metric distances, the planning on the road graph is sufficient to provide accurate navigation information, e.g. "Take the next left". In an indoor setting, it may be sufficient to localize to within a room. Topological mapping is sufficient if it provides reliable and correct localization.

It is important to distinguish spatial non-metric from non-spatial. A topological map still contains spatial information, but is not tied to an absolute uniform scale. Relative distances, circular ordering, spatial ordering and more incorporate spatial knowledge without resorting to metric information. Scene *classification* is able to provide semantic labeling for a room or scene, but is unable provide spatial information between these same rooms or scenes.

Finally, a semantic system is one where human semantics play a primary role. Foremost, a semantic representation enables input from human users, and provides output meaningful to a human user. Additionally, incorporating human semantics into the reasoning provides a more intuitive system. Increased understanding provides both more insight, and allows for more informed operation by a human operator. On the other hand, injecting human semantics potentially increases the complexity and may add unnecessary indirection.

1.3 Contributions

This dissertation advances knowledge of semantic mapping and localization in three ways while satisfying the constraints listed in the previous section. Chapter 2 will review past and present approaches and cover background material. In chapter 3, a method for creating

compact topological maps from video sequences is presented. The resulting maps represent groupings of multiple frames into unique spatial locations. A novel image similarity score is computed from a cyclical alignment of sorted feature sequences to incorporate weak geometric verification. Temporal semantics are introduced in an MRF lattice to detect loop-closures. Finally the images are clustered into nodes using temporal information to yield the final topological map. This map bypasses all metric consideration while using only visual information.

A novel localization approach is detailed in chapter 4 where objects are employed in lieu of more traditional features in order to localize a robot. Instead of relying on discrete localization of objects within images, heatmaps are used to weakly estimate the presence or absence of objects in an image. This in turn is used to grade the feasibility of the estimated position. By using a particle filter it's possible to sample the space of possible locations, and over time estimate the location of the camera. By directly using human defined objects, this localization incorporates human semantics directly.

The complement is explored in chapter 5 where the layout of objects is recovered from a known location. While chapter 4 dealt with human output for machine use, this chapter examines automatically creating human meaningful representations of the environment.

1.3.1 Published work supporting this thesis

The work on topological map creation with MRF loop-closure detection (chapter 3) was first published in Anati and Daniilidis [2009]. Using object detections for localization (chapter 4) was published in Anati et al. [2012].

Chapter 2

Background and Related Work

Semantic mapping and localization has garnered a lot of attention in recent years. The following sections cover a selection of past and recent work in the field. Each section mirrors one of the contributions, with topological mapping surveyed in section 2.1. This is followed by a survey of semantic localization in section 2.2 and finally object localization in section 2.3.

2.1 Topological Mapping

As specified earlier, topological mapping is a major sub-problem within semantic mapping. Although research in topological mapping is varied [Garcia-Fidalgo and Ortiz, 2015b], the underlying goal is always the construction of the graph-based map to represent the environment. An overview of related works of relevance to topological mapping can be found in table 2.1. Every row is categorized for the properties expected of a desirable solution as set forth in section 1.1.1:

Non-Metric: Map doesn't require measuring absolute *physical* distances between nodes.

Pure Vision: Visible-light cameras are the only sensor used (no time-of-flight).

Spatial: The underlying graph represents spatial relationships between places (definition 1.1).

Semantic: Map incorporates or is built with semantics.

A few select works will be expanded on next.

Table 2.1: Survey of work in Topological Mapping

	Non-Metric	Pure Vision	Spatial	Semantic
Choset et al. [2000]	✘	✘	✔	✔
Choset and Nagatani [2001]	✘	✘	✔	✔
Bosse et al. [2003]	✘	✘	✔	✘
Tomatis et al. [2003]	✘	✘	✔	✘
Modayil et al. [2004]	✘	✘	✔	✘
Beeson et al. [2005]	✘	✘	✔	✔
Limketkai et al. [2005]	✘	✘	✔	✔
Tapus and Siegwart [2005]	✔	✘	✔	✘
Zivkovic et al. [2005]	✔	✔	✔	✘
Ranganathan et al. [2006]	✔	✘	✔	✘
Fraundorfer et al. [2007]	✔	✔	✔	✘
Goedemé et al. [2007]	✔	✔	✔	✘
Ho et al. [2007]	✘	✘	✔	✔
Angeli et al. [2008]	✘	✔	✔	✘
Konolige and Agrawal [2008]	✘	✔	✔	✘
Oberlander et al. [2008]	✘	✘	✔	✔
Booij et al. [2009]	✔	✔	✔	✘
Klein and Murray [2007, 2009]	✘	✔	✔	✘
Liu et al. [2009]	✔	✔	✔	✘
Murillo and Kosecka [2009]	✘	✔	✔	✘

Table 2.1: Survey of work in Topological Mapping (Continued)

	Non-Metric	Pure Vision	Spatial	Semantic
Cummins and Newman [2010b]	✘	✔	✔	✘
Singh and Košecká [2010]	✔	✔	✔	✘
Erkent and Bozma [2012]	✔	✔	✘	✘
Lui and Jarvis [2012]	✘	✔	✔	✔
Singh and Kosecka [2012]	✔	✔	✔	✔
Latif et al. [2014]	✔	✔	✔	✘
Liu et al. [2014]	✘	✘	✔	✔
Rituerto et al. [2014]	✘	✔	✔	✔
Volkov et al. [2015]	✔	✔	✘	✘
Garcia-Fidalgo and Ortiz [2015a]	✔	✔	✔	✘
Kejriwal et al. [2016]	✔	✔	✘	✘
Korrapati and Mezouar [2016]	✔	✔	✔	✔

The generalized Voronoi graph (GVG) formulation of Choset et al. [2000], defines a topological map as the set of points equidistant to two or more obstacles. Beeson et al. [2005] further develop this with reduced extended Voronoi graphs (REVG) specifically to address those cases where GVG fail (L-junctions, large rooms). Both approaches use a 2D occupancy grid of obstacles as their input presumable provided by some depth sensor such as LIDAR.

Both the Atlas framework [Bosse et al., 2003] and Tomatis et al. [2003] utilize a hybrid approach that combines metric and topological features for indoor mapping and localization. In Bosse et al. [2003] nodes represent local metric map frames within a topological graph network. Spatially *overlapping* map frames are connected by probabilistic edges denoting transformation between them and their uncertainties. The size of local maps is bounded, yielding an upper bound on metric processing. In contrast, the work of Tomatis

et al. [2003] uses both metric and topological nodes within their global graph. The metric nodes map rooms and are disconnected, allowing for precise metric navigation while topological nodes are used for hallways where simpler planning is sufficient. Since the local maps do not overlap, loop-closure is only addressed at the global graph by monitoring divergence in a POMDP algorithm. In both of the above, there is no vision component. Bosse et al. [2003] use LIDAR and sonar generate local metric maps while Tomatis et al. [2003] employ LIDAR only.

FAB-MAP [Cummins and Newman, 2008a,b] and FAB-MAP 2.0 [Cummins and Newman, 2009, 2010a,b] use visual words of local image features to describe locations. Each incoming image is localized to the existing map. If localization determines a new place has been observed, the map is updated with a new node and appearance model. The localization itself is performed by using recursive Bayes. Accuracy is improved by modeling pairwise correlations between words. This reduces the impact of detecting highly correlated visual words.

In Singh and Kosecka [2012], semantic segmentation of street level panoramas is used to accomplish two tasks. Segments in each panorama are labeled using semantic categories such as sky, building, tree. Features generated from this segmentation are used to cluster the panoramas into different locations creating a topological map. In addition the "semantic label descriptor" is used train a scene classifier for recognizing road intersections in an urban setting.

Recently, Latif et al. [2014] treat loop-closing as a sparse optimization problem. Descriptors are computed for incoming images. A dictionary of previously seen images is used to form a basis to decompose the current image. The problem is regularized to minimize the number of basis images used in the reconstruction. The result is an approximation which matches the current image to the closest previously seen image.

Coresets are used to define locations by Volkov et al. [2015]. Video stream segments are embedded into a sub-space. The embedding is then approximated using sets of piecewise linear functions called coresets. Coresets are structured in a tree allowing for a

coarse-to-fine search. Finally, new images are matched to existing coresets via Naive Bayes.

None of these approaches meet all the requirements set forth in section 1.1.1 with the exception of Singh and Kosecka [2012]. Both Bosse et al. [2003]; Tomatis et al. [2003] are not purely topological, containing local metric maps. On the other hand, Beeson et al. [2005]; Choset et al. [2000] do not use any visual data. Volkov et al. [2015] is purely topological, but does not contain any spatial information. Its resulting maps contain no edges; the nodes exist in complete isolation from each other. Finally, Cummins and Newman [2008a,b, 2009, 2010a,b]; Latif et al. [2014] do not employ semantics. Images are never clustered into *places* and semantic image content isn't used.

2.2 Navigation and Localization

The driving force behind semantic mapping is its applications. The primary motivation for creating maps is navigation and localization. Work reviewed here is primarily focused on the localization and navigation tasks. However, several combine both navigation and mapping ([Bosse et al., 2003; Fraundorfer et al., 2007; Klein and Murray, 2007; Tomatis et al., 2003]). Those that don't assume some prior map is provided. An overview of the literature can be found in table 2.2. In addition to the properties mentioned in section 1.1.2, *automation* is also added:

Pure Vision: Visible-light cameras are the only sensor used (no time-of-flight).

Semantic: Localization relies on semantics (definition 1.3).

Localization: Position is estimated with respect to a map: pre-existing or newly built.

Automated: Position is estimated without human intervention.

Table 2.2: Survey of work in Semantic Localization

	Pure Vision	Semantic	Localization	Automated
Bosse et al. [2003]	✘	✘	✔	✔
Tomatis et al. [2003]	✘	✘	✔	✔
Rottmann et al. [2005]	✘	✔	✔	✔
Ekvall et al. [2006]	✘	✔	✔	✔
Booij et al. [2007]	✔	✘	✔	✔
Fraundorfer et al. [2007]	✔	✘	✔	✔
Valgren and Lilienthal [2007]	✔	✘	✔	✔
Scaramuzza et al. [2008]	✔	✘	✔	✔
Klein and Murray [2007, 2009]	✔	✘	✘	✔
Espinace et al. [2010]	✘	✔	✘	✔
Liu et al. [2010]	✘	✘	✔	✔
Murillo et al. [2010]	✔	✘	✔	✔
Pronobis et al. [2010a,b,c]	✘	✔	✘	✔
Steder et al. [2010]	✘	✘	✔	✔
Zamir and Shah [2010]	✔	✘	✔	✔
Apostolopoulos et al. [2012]	✘	✔	✔	✘
Sattler et al. [2012]	✘	✘	✔	✔
Bader et al. [2013]	✔	✘	✔	✔
Botterill et al. [2013]	✔	✘	✔	✔
Murillo et al. [2013]	✔	✘	✔	✔
Salas-Moreno et al. [2013]	✘	✔	✔	✔
Bader and Vincze [2014]	✔	✘	✔	✔
Drouilly et al. [2014]	✔[1]	✔	✔	✔
Ivanov et al. [2015]	✘	✘	✔	✔

[1] Position *refined* using depth channel.

Table 2.2: Survey of work in Semantic Localization (Continued)

	Pure Vision	Semantic	Localization	Automated
Kanji [2015a,b]	✔	✘	✘	✔
McManus et al. [2015]	✔	✘	✔	✔
Ribeiro et al. [2015]	✘	✔	✔	✔
Atanasov et al. [2014, 2016]	✔	✔	✔	✔

Pronobis et al. [2010a,c] combine both laser and cameras to classify rooms. In addition to geometric properties, laser data is used also to approximate the shape of the room. Both global and local image features are extracted using the camera and used for visual classification. A discriminative classifier is trained for each cue, and these cues are combined using a SVM to create a multi-model classifier.

The indoor localization approach of Liu et al. [2010] also uses a combination of global and local image features augmented with an IMU. Yaw information from the IMU is used to filter out reference frames which are incompatible with the current query image. The global image feature, in this case GIST [Oliva and Torralba, 2001], is used to select an initial set of candidates. SURF features [Bay et al., 2006] are used to match the query to the candidates. To improve robustness, actual localization is performed using a Hidden-Markov-Model (HMM) to integrate cues over time, and expand the candidate pool via a simple motion model.

In contrast, Murillo and Kosecka [2009] use GIST for outdoor localization. A codebook of GIST descriptors is learned using K-means clustering of GIST features computed over a reference set. An input panorama is divided into four sections, and GIST codewords are computed for each section. The cyclic sequence of codewords is used to perform preliminary matching and alignment to candidates from the map. Final scores are calculated using Euclidean distance of aligned GIST features.

The extraction vertical lines from omnidirectional images form the basis of image matching in Scaramuzza et al. [2008]. Lines are extracted using the Sobel filter. Gradient histograms are computed at different positions along detected lines and used to define a descriptor. These descriptors are used to match vertical lines between images enabling tracking across frames. The resulting tracks are fed to an EKF-SLAM algorithm resulting in localization and mapping.

In Espinace et al. [2010] scenes are recognized solely through object detections. Object classifiers are trained from image datasets using a combination of image features (e.g. gradients, intensity values) and range data (height from ground plane, object dimensions and depth dispersion). Classifier confidence is estimated from training data and the configuration of objects in different scenes is learned from images and does not consider any appearance information, yielding a semantic scene classifier. At query time, a sliding window is used to densely search an image. 3D properties are used initially to prune the list of candidate windows, followed by classification using the visual features. Once objects are detected, the probability of each scene category is computed.

Apostolopoulos et al. [2012] create an interactive navigation system designed to guide a user to a target location. The system assumes a annotated map with landmarks and is given the user's initial location. An optimal path is computed to the target, and a set of directions is fed to the user. At each step the user confirms both the presence of expected landmarks and the execution of provided directions. A particle filter is used to maintain the location of the user, and is updated both by user feedback and sensors (e.g. compass, pedometer).

Instead of a map, Zamir and Shah [2010] use a set of GPS-tagged images taken from StreetView. SIFT descriptors of the reference images are stored in an efficient ANN (approximate nearest neighbor) structure for fast retrieval. Irrelevant image descriptors are removed from consideration using a novel ratio test which is based on the GPS position of a reference image. Instead of localizing single images, the system localized a group of

query images taken in proximity of each other. First, each query image is localized independently to generate a set of candidate location. For each candidate location, a subset of the database is selected based on proximity to the candidate and matched against the entire query set. The candidate set with the best matches is used to label each query image.

SLAM++ [Salas-Moreno et al., 2013] combines semantic localization and mapping of a scene by tracking the camera's position three or more objects. 3D meshes of the scene are extracted using depth sensors. Predefined object meshes are then used to localize objects in the scene using dense ICP. An error defined over the graph of both objects and camera poses is optimized using a least squares solver.

A complex three layer map is used by Drouilly et al. [2014] for localization. Their Metric-Topological-Semantic Map (MTS Map) combines RGDB data (metric) with scene categorization (semantic) and scene geometry (topological) to define local clusters of similar images. These hybrid maps are then used for localization by matching geometry of structure of the current scene with the map.

Recent work by Atanasov et al. [2014, 2016] use objects detected in imagery to localize the robot in a prior map. Potential errors due to missed and false detections are modeled combinatorically. This allows the system to find the likeliest position given all possible associations between objects in the map and observed detections.

The majority of these approaches contain no semantics of any sort [Liu et al., 2010; Murillo and Kosecka, 2009; Scaramuzza et al., 2008; Zamir and Shah, 2010]. Although semantic, Espinace et al. [2010]; Pronobis et al. [2010a,c] solve the scene *classification* problem, and tackle localization as an afterthought (in the case of Espinace et al. [2010] not at all). Apostolopoulos et al. [2012] is semantic, it is far from automated requiring constant user interaction. Only Atanasov et al. [2014, 2016] combines all the requisite properties desired of a semantic localization system.

2.3 Object Detection, Recognition and Localization

Another class of mapping algorithms focus instead on recovering objects and landmarks in a scene. In addition to manipulation and grasping, object detection, recognition and localization are useful for such tasks as scene classification [Rottmann et al., 2005] or localization [Apostolopoulos et al., 2012]. Included are both image-based localization [Li et al., 2010b; Torresani et al., 2010; Yao et al., 2012] which incorporate semantics, and scene-based approaches [Anguelov et al., 2002; Bao et al., 2012; Castle and Murray, 2009; Lai et al., 2012]. A sampling of the literature can be found in table 2.3. These include other *semantic mapping* approaches that do not use objects directly, but incorporate semantics in other ways [Kostavelis and Gasteratos, 2013; Rituerto et al., 2014; Walter et al., 2013]. Finally Liu et al. [2016] provide of survey of *semantic mapping* approaches, both with and without objects. Every row is categorized for the properties expected of a desirable solution as set forth in section 1.1.3:

Pure Vision: Visible-light cameras are the only sensor used (no time-of-flight).

Object Type: What kind of objects does the system model.

Object Location: Are objects localized in a map.

Object Pose: Is the pose of the object recovered.

Li2010a

Table 2.3: Survey of work in Semantic Mapping

	Pure Vision	Object		
		Types	Location	Pose
Anguelov et al. [2002]	✘	Solid	✔	✔
Ekvall et al. [2006]	✔	Flat	✔	✘
Castle et al. [2007]	✔	Flat	✔	✘

Table 2.3: Survey of work in Semantic Mapping (Contintued)

	Pure Vision	Object		
		Types	Location	Pose
Lopez et al. [2008]	✘	Flat	✔	✘
Meger et al. [2008]	✘	Any	✔	✘
Zender et al. [2008]	✘	Any	✔	✘
Bao et al. [2011b, 2010]	✔	Any	✔	✔
Castle et al. [2010]	✔	Flat	✔	✔
Douillard et al. [2010]	✘	Any	✔	✘
Li et al. [2010b][3]	✔	Any	✘	✘
Pronobis et al. [2010c]	✔	Any	✘	✘
Torresani et al. [2010][3]	✔	Any	✘	✘
Arbeiter et al. [2011]	✘	Table-tops	✔	✔
Bao and Savarese [2011]	✔	Any	✔	✔
Case et al. [2011]	✘	Text	✔	✘
Civera et al. [2011]	✔	Any	✔	✔
Lai et al. [2011a]	✘	Any	✘	✔
Varadarajan and Vincze [2011]	✔[2]	Walls & Doors	✔	✘
Zhou et al. [2011]	✔[2]	Any	✔	✔
Bao et al. [2012]	✔	Any	✔	✔
Lai et al. [2011a][3]	✘	Any	✘	✔
Pronobis and Jensfelt [2012]	✔	Any	✘	✘
Ko et al. [2013]	✔	Flat	✔	✘
Kostavelis and Gasteratos [2013]	✘	None[4]	✘	✘
Salas-Moreno et al. [2013]	✘	Any	✔	✔

[2]Use a calibrated stereo camera pair.
[3]Not a mapping approach.

Table 2.3: Survey of work in Semantic Mapping (Contintued)

	Pure Vision	Object		
		Types	Location	Pose
Sengupta et al. [2013]	✔[2]	Any	✔	✘
Walter et al. [2013]	✘	None[4]	✘	✘
Lai et al. [2014]	✘	Any	✘	✘
Rituerto et al. [2012, 2014]	✔	None[4]	✘	✘
Cleveland et al. [2015]	✘	Any	✔	✔
Vineet et al. [2015]	✔[2]	Any	✘	✘
Wong et al. [2013, 2015]	✘	Any	✔	✔

2D planar objects are localized in a 2D maps by Lopez et al. [2008]. The robot explores a single room given both a metric and topological map of a room. The occupancy map is divided into squares, and the robot exhaustively searches for objects in every square. Using images captured from the camera, histograms of patches are computed and used to initialize object hypothesis. Hypothesis are refined by iteratively estimating the depth of the object and capturing new images. Final hypothesis are verified using SIFT matching [Lowe, 2004].

Castle et al. [2010] accomplish recognition and localization of *planar* objects in 3D. Their key insight is to use the same features already tracked for camera localization to detect and localize known objects. A database of planar objects is first constructed using SIFT keypoints. For camera localization they use the PTAMM framework [Castle et al., 2008]. When at least three keypoints are detected and triangulated, they are used to recover a 3D plane using RANSAC. Since only planar objects are considered, detection is performed by only matching database keypoints to plane keypoints. By randomly testing pairs of matching points the final position, scale and orientation of the object is recovered.

[4]Places are semantically annotated, but without the use of objects.

Bao et al. [2010] is similar in that it estimates both objects and planes, but is able to handle 3D objects. The position and orientation of the objects and supporting planes is estimated simultaneously. Objects are assumed to lie flat on a set of parallel planes (i.e. object *up* vector matches plane normal). Size of candidate object detections is used to estimate distances to the planes, while object poses provide information regarding plane normals. The planes' normal is estimated by solving a linear system which assumes at least three objects are not colinear in the image. An exhaustive search of the solution space is performed to find the optimal solution. The search time is reduced by employing parallel computing techniques.

Semantic Structure from Motion [SSFM Bao et al., 2012] localizes points, regions and objects simultaneously in 3D. The primary advantage of this approach is verification through interaction using points on objects / regions, and objects on regions. An estimated plane (region) in 3D is expected to have the normal of an object perceived to stand on that surface. Objects are initially detected in images using an off-the-shelf detector and matched across images using back-projection.

RGB-D sensors are used by both OP-Trees [Object-Pose Trees Lai et al., 2011a] and Lai et al. [2012]. OP-Trees do not solve the detection problem, but focus instead on recognition and pose. An OP-Tree is a tree of object classifiers divided into four level: category, instance, view and pose. A detection is processed by each level of the tree in sequence, until the category, instance view and pose of the object is recovered. The underlying classifiers use a combination of RGB and depth features computed over image patches. On the other hand, Lai et al. [2012] generate a 3D point cloud of the environment using multiple images. Each RGB-D frame is processed using a view based object detector to recover a detection score for each pixel. Using the depth information, each pixel score is back-projected to the source voxel in the environment. Voxel scores are then integrated across all input frames to account for multiple sightings of the same voxel. These are incorporated into a graph framework with a smoothness constraint and solved using graph cuts.

A key limitation of both Castle and Murray [2009] and Lopez et al. [2008] is that they are both limited to planar objects. As such they are unable to deal with solid objects scattered about an environment. By relying only on LIDAR, only non-contiguous objects can be detected. Objects touching the wall are labeled as part of the wall, and objects touching each other are grouped into single entities. Similarly Lai et al. [2012] relies on recovered depth information from an RGB-D camera to localize and segment objects. The work of Lai et al. [2011a] is able to recover object class and pose, but relies on prior detections and does not perform localization. Both Bao et al. [2010] and Bao et al. [2012] incorporate semantics in the localization step, but are bound to whatever initial detections are fed to it at initialization and are unable to adjust beyond small variations in position and pose.

2.3.1 Semantic Object Detection

In addition to the methods outlined above, a subset of recognition and detection algorithms employ semantics in their training stage. These generally include some form of heatmaps, a dense response to filters or detectors across entire images. Even though they do not map environments, these works bear special interest as they not only produces semantic output, but also employ semantic reasoning.

Torresani et al. [2010] introduce *classemes* to create an object detection system capable of dealing with unknown object classes. Classemes are mid-level features designed to detect properties of objects and relationships between objects. Different classemes are learned by training classifiers on low-level image features such as HOG (Histogram of Gradients) or GIST in a supervised manner. Responses to the classeme classifiers are then used as features for learning higher level object detectors.

In Object Bank [Li et al., 2010b], images are represented by a set of dense responses to a bank of object detectors. Responses for object detector is binned using a spatial pyramid at multiple scales to create an object response histogram. The OB (Object Bank) representation is used to perform scene classification on both indoor and outdoor images.

Instead of using a bank of object detectors, Yao et al. [2012] learn a set of template detectors. Training images are randomly sampled to generate an extremely large set of templates. Dense responses to these templates are used as image features to train a discriminative classifier. Additional constraints are added to the optimization to reduce correlation between classifiers, eliminating redundant templates.

These approaches are novel in their application of semantics in the detection process. Both classemes and random templates [Yao et al., 2012] are not required to have semantic meaning but implicitly do. On the other hand, Object Bank [Li et al., 2010b] explicitly incorporates semantics. At this time, these approaches are still limited to pixel based localization and cannot be used in a robotic setting.

Chapter 3

Topological Map with MRF Loop-Closure Detection

3.1 Introduction

The first challenge discussed in chapter 1, topological mapping, has gained prominence in recent years. Their importance is two-fold as topological maps are both more robust than metric maps and cognitively more plausible for use by humans. By representing an environment as a graph, the topological map imposes a network on the space. This creates bounded discrete representations of continous environments.

Topological maps are used to perform path planning, by providing way-points, and defining reachability of places. They also serve to correct for the drift in visual odometry systems and can be part of hybrid representations where the environment is represented metrically locally but topologically globally.

This chapter presents a novel system for constructing topological maps from video sequences. Recall definition 1.1, where a topological map is a graph, $T = (V, E)$, where V is a set of locations and E are edges describing connectivity between locations. For the map to be useful, it needs to incorporate loop-closures.

Loop-closure For any two locations $i, j \in V$, E contains the edge (i, j) if and only if

it is possible to reach location j from location i without passing through any other location $k \in V$.

Additionally, in the context of creating such maps from video, there are two more desirable properties:

Spatial distinctiveness Two images from "different locations" *must not* be represented by the same graph node.

Compactness Two images taken at the "same location" *should* be represented by the same graph node.

Note that spatial distinctiveness *requires* distinguishing between separate locations, however compactness merely *encourages* clustering of spatially similar images. This distinction is important, as lack of compactness does not lead to errors in either path planning or visual odometry while breaking spatial distinctiveness does.

Following this, the construction of topological maps can be broken down into two tasks:

- Determining whether two images have been taken from the same place (E), and
- Reducing the original set of video frames to a smaller representative set of nodes (V).

Accurately determining whether two images are taken from the same location reflects both on loop-closure and spatial distinctiveness. If two images taken in the same location are not detected as such then a possible loop-closure is missed. On the other hand, if two images from different locations are incorrectly matched then they will be represented by the same node, breaking spatial distinctiveness. While reducing the original set of video frames to a smaller set of nodes clearly affects compactness, it also affects loop-closure. When two images that should be represented by the same node do not get clustered together, it potentially creates a situation with a triangle i, j, k such that i is connected to j, but to reach location j from location i requires traversing k.

Accomplishing these tasks requires meeting a number of challenges. Foremost among these is the problem of *perceptual aliasing* [Cummins and Newman, 2008a], where different locations have similar appearance (e.g. figure 3.7a and figure 3.7b). These result in incorrect matches, creating incorrect loop-closures leading to a map which does not accurately represent the environment. The opposite problem, *perceptual variability*, when two images taken at the same location are visually dissimilar, prevents loop-closures from being correctly detected. In addition to the problems of perception, scalability is also an issue. Determining whether two images were captured at the same location could potentially require processing every pair of images. Given a video sequence of thousands or tens of thousands of images this becomes a very time consuming process.

The proposed approach focuses primarily on the challenges of *perceptual aliasing* and *perceptual variability* and incorporates the following three innovations to topological mapping:

1. A novel image similarity score which uses dynamic programming to match images using both the appearance and the layout of the features in the environment.

2. The use of graphical models to detect locally consistent loop-closures.

3. Utilizing the temporal assumption to generate compact topological maps using minimum dominating sets.

This is divided into three modules: calculating image similarity, detecting loop closures, and map construction. As defined, it is possible to implement each module independently, providing great flexibility in the algorithm selection. What follows is the definition of the interfaces between each pair of modules.

Starting with $\mathcal{I}$, a sequence of n images, the result of calculating image similarity scores is a matrix $M_{n\times n}$ where M_{ij} represents a relative similarity between images i and j. Section 3.3 describes how local image features are used to compute the matrix M. In order to detect loop-closures, M must be discretized into a binary decision matrix $D_{n\times n}$ where $D_{ij} = 1$ indicates that images i and j are spatially equivalent representing the same

"location" and thus form a loop closure. The construction of the matrix D is explained in section 3.4 and is achieved by defining a Markov Random Field (MRF) on M and perform approximate inference using Loopy Belief Propagation (Loopy-BP). In the final step, the topological map T is generated from D. A set of nodes V and their associated connectivity E is calculated in section 3.5 using the minimum dominating set of the graph represented by D.

3.2 Related Work

The state of the art in topological mapping of images is the FAB-MAP [Cummins and Newman, 2008a] system. FAB-MAP uses bag of words (BoW) to model locations using a generative appearance approach that models dependencies and correlations between visual words rendering FAB-MAP extremely successful in dealing with the challenge of perceptual aliasing. Its implementation outperforms any other in speed averaging an intra-image comparison of less than 1ms. FAB-MAP 2.0 improved performance by reducing the number of intra-image comparisons executed, Cummins and Newman [2009, 2010b] by including inverted-index search and Cummins and Newman [2010a] by accelerating the likelihood calculation using an early "bail-out" strategy.

Bayesian inference is also used in Angeli et al. [2008] where bags of words on local image descriptors model locations whose consistency is validated with epipolar geometry. Ranganathan et al. [2006] incorporate both odometry and appearance and maintain several hypotheses of topological maps. Older approaches like ATLAS [Bosse et al., 2003] and Tomatis et al. [2003] define maps on two levels, creating global (topological) maps by matching independent local (metric) data and combining loop-closure detection with visual SLAM. The ATLAS framework [Bosse et al., 2003] matches local maps through the geometric structures defined by their 2D schematics whose correspondences define loop-closures. Tomatis et al. [2003] detect loop closures by examining the modality of the robot position's density function. A density function with two modes traveling in sync is

the result of a missed loop-closure, which is identified and merged through backtracking.

Approaches like Booij et al. [2007]; Fraundorfer et al. [2008]; Valgren et al. [2006, 2007] represent the environment using only an image similarity matrix. Booij et al. [2007] use the similarity matrix to define a weighted graph for robot navigation. Navigation is conducted on a node by node basis, using new observations and epipolar geometry to estimate the direction of the next node. Valgren et al. [2006] avoid exhaustively computing the similarity matrix by searching for and sampling cells which are more likely to describe existing loop-closures. While in Valgren et al. [2007] spectral clustering is used to reduce the search space incrementally as new images are processed. Fraundorfer et al. [2008] use hierarchical vocabulary trees [Nister and Stewenius, 2006] to quickly compute image similarity scores. They show improved results by using feature distances to weigh the similarity score. Goedemé et al. [2007] propose 'invariant column segments' combined with color information to compare images. This is followed by agglomerative clustering of images into locations. Potential loop-closures are identified within clusters and confirmed using Dempster-Shafer probabilities.

Latif et al. [2014] present an online incremental approach to loop-closing that doesn't require offline training of image descriptors. A descriptor is computed for each new image, and represented using a basis formed from the descriptors of all previous images. The sparse optimization finds this representation, and the image with the most significant contribution (largest coefficient) is deemed the loop-closure candidate.

Semantic segmentation is used by Singh and Kosecka [2012] to detect loop-closures. Image regions are labelled with semantic categories (e.g. building, street) and combined with local features to describe locations. These are then clustered to generate a topological map. K-means clustering of descriptors prunes the underlying representation.

The approach described in this chapter advances the state of the art by using a powerful image alignment score without employing full epipolar geometry, and robust loop closure detection by applying MRF inference on the similarity matrix. Both the MRF and the clustering step incorporate temporal semantics. Together with [Booij et al., 2005], it is

the only video-based approach that provides a greatly reduced set of nodes for the final topological representation, making thus path planning tractable.

3.3 Image Similarity Score

The first step is calculating the intra-image similarity score. For any two images i and j, the similarity score M_{ij} is computed in three steps: generating image features, sorting image features into sequences, and finding the optimal alignment between both sequences. To detect and generate image features Scale Invariant Feature Transform (SIFT) [Lowe, 2004] are used. SIFT was selected as it is invariant to rotation and scale, and partially immune to other affine transformations.

3.3.1 Feature sequences

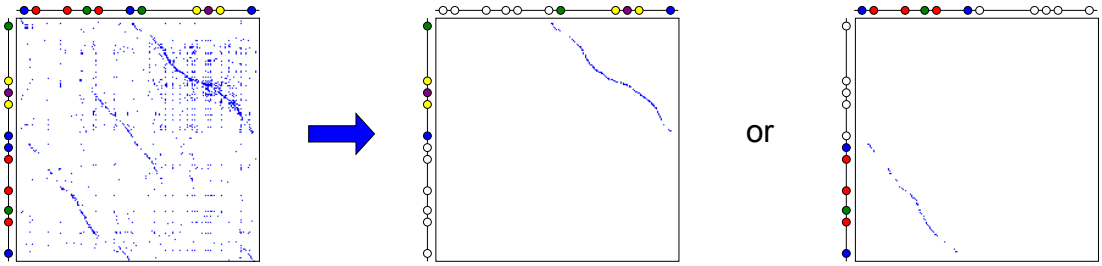
Simply matching the SIFT features by value [Lowe, 2004] yields very positive results (see figure 3.3) but they do not account for perceptual aliasing. Bearing in mind that image features represent real world structures with fixed spatial arrangements a robust similarity score would benefit by taking into account their relative positions. A popular approach, employed in [Cummins and Newman, 2009; Tardif et al., 2008], is to enforce scene rigidity by validating the epipolar geometry between two images. This process, although extremely accurate, is expensive and very time-consuming. Instead, the only geometric information used from the SIFT features is their bearing with respect to the camera. Thus geometric consistency is reduced from two dimensional feature positions to one dimensional ordering of feature sequences. The feature sequences are generated by calculating spherical coordinates of each feature, discarding the elevation component and sorting features by bearing only. Image similarity is calculated by searching for an optimal alignment between pairs of sequences, this incorporates both the SIFT descriptor and relative spatial order of SIFT features into our similarity score.

3.3.2 Sequence alignment

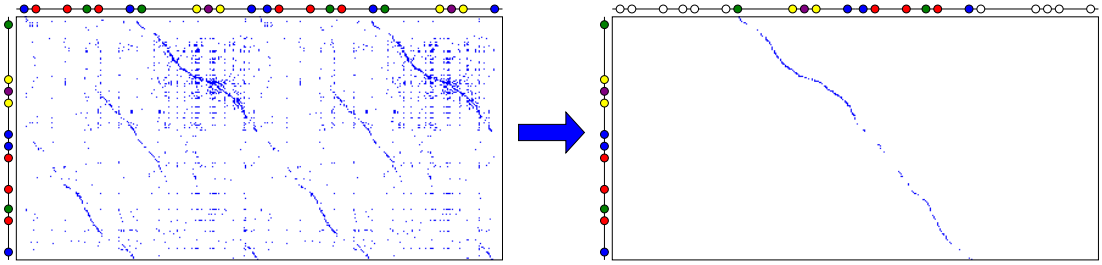
To solve for the optimal alignment between two ordered sequences of features we employ dynamic programming. Here a match between two features, f_a and f_b, occurs if their L_1 norm is below a threshold, $Score(a, b) = 1$ if $|f_a - f_b|_1 < t_{match}$. A key aspect to dynamic programming is the enforcement of the ordering constraint. This ensures that the relative order of features matched is consistent in both sequences, exactly the property desired to ensure consistency between two scene appearances. Since bearing is not given with respect to an absolute orientation, ordering is meant only cyclically, which can be handled easily in dynamic programming by replicating one of the input sequences (see Figure 3.1). Modifying the first and last rows of the score matrix to allow for arbitrary start and end locations yields the optimal cyclical alignment in most cases. This comes at the cost of allowing one-to-many matches which can result in incorrect alignment scores. The score of the optimal alignment between both sequences of features provides the similarity score between two images and the entries of the matrix M. The values of M_{ij} are calculated for all $i < j - w$. Here w represents a window used to ignore images immediately before/after the query image. For two sequences of features $F_1 = (f_1^1, \ldots, f_1^n)$ and $F_2 = (f_2^1, \ldots, f_2^m)$, for images i' and j' respectively, the optimal alignment A is found. Where $A = (A_1, A_2)$ and A_1, A_2 are indicator vectors for which features in F_1 matches to those in F_2 ($A_1 \in \{0, 1\}^n$, $A_2 \in \{0, 1\}^m$ and $\sum A_1 = \sum A_2 = k$). The score of the alignment is $Score(A) = s_{match}k - s_{gap}(n - k) - s_{gap}(m - k)$.

3.4 Loop-Closure Detection Using MRF

Using the image similarity measure matrix M, Markov Random Fields are used to detect loop-closures. A lattice H is defined as an $n \times n$ lattice of binary nodes where a node $v_{i,j}$ represents the probability of images i and j forming a loop-closure. The matrix M provides an initial estimate of this value. The factor $\phi_{i,j}$ is define for the node $v_{i,j}$ as follows: $\phi_{i,j}(1) = M_{ij}/F$ and $\phi_{i,j}(0) = 1 - \phi_{i,j}(1)$ where $F = \max(M)$ is used to



(a) Sequence Alignment with Standard Dynamic Programming



(b) Sequence Alignment with Cyclically Invariant Dynamic Programming

Figure 3.1: Using cyclically invariant dynamic programming to account for changes in robot direction. Standard dynamic programming (a) fails to find the correct alignment whereas (b) finds the correct alignment using a copy of the cost matrix.

normalize the values in M to the range $[0, 1]$. Loops closures in the score matrix M appear as one of three possible shapes. In an intersection the score matrix contains an ellipse. A parallel traversal, when a vehicle repeats part of its trajectory, is seen as a diagonal band. An inverse traversal, when a vehicle repeats a part of its trajectory in the opposite direction, is an inverted diagonal band. The length and thickness of these shapes vary with the speed of the vehicle (see figure 3.2 for examples of these shapes). Therefore the lattice H uses eight way connectivity, as it better captures the structure of possible loop closures.

As adjacent nodes in H represent sequential images in the sequence, significant overlap between them can be expected. Two neighboring nodes (in any orientation), are expected to have similar scores. Sudden changes occur when either a loop is just closed (sudden increase) or when a loop closure is complete (sudden decrease) or due to noise

(a) Intersection (b) Parallel Traversal (c) Inverse Traversal

Figure 3.2: A small ellipse resulting from an intersection (a) and two diagonal bands from a parallel (b) and inverse (c) traversals. All extracted from a score matrix M.

caused by a sudden occlusion in one of the scenes. Imposing smoothness on the labeling captures loop closures while discarding noise. Edge potentials are therefore defined as Gaussians of differences in M. Letting $G(x, y) = e^{-\frac{(x-y)^2}{\sigma^2}}$, $k = \{i-1, i, i+1\}$ and $l = \{j-1, j, j+1\}$ then

$$\phi_{i,j,k,l}(0,0) = \phi_{i,j,k,l}(1,1) = \quad \alpha \cdot G\left(M_{ij}, M_{kl}\right) \tag{3.1}$$

$$\phi_{i,j,k,l}(0,1) = \phi_{i,j,k,l}(1,0) = \qquad 1, \tag{3.2}$$

where $1 \leq \alpha$ (we ignore the case when both $k = i$ and $j = l$). Overall, H models a probability distribution over a labeling $v \in \{1, 0\}^{n \times n}$ where:

$$P(v) = \frac{1}{Z} \prod_{i,j \in [1,n]} \phi_{i,j}(v_{i,j}) \prod_{i,j \in [1,n]} \prod_{k=[i-1,i+1]} \prod_{l=[j-1,j+1]} \phi_{i,j,k,l}(v_{i,j}, v_{k,l}) \tag{3.3}$$

In order to solve for the MAP labeling of H, $v^* = \arg\max_v P(v)$, the lattice must first be transformed into a cluster graph C. This transformation models the beliefs of all factors in the graph and the messages being passed during inference. Every node and every edge in H is modelled as a node in the cluster graph C. An edge exists between two nodes in the cluster graph if the relevant factors share variables. In addition this construction presents a two step update schedule, alternating between 'node' clusters and 'edge' clusters as each class only connects to instances of the other. Once defined, a straightforward

implementation of the generalized max-product belief propagation algorithm (described in both [Bishop, 2006] and [Koller and Friedman, 2009]) serves to approximate the final labeling. The cluster graph is initialized directly from the lattice H with $\psi_{i,j} = \phi_{i,j}$ for nodes and $\psi_{i,j,k,l} = \phi_{i,j,k,l}$ for edges. The MAP labeling found here defines our matrix D determining whether two images i and j close a loop. Note, that the above MAP labeling is guaranteed to be locally optimal, but is not necessarily consistent across the entire lattice. Generally, finding the globally consistent optimal assignment is NP-hard [Koller and Friedman, 2009]. Instead, we rely on our definition of D, which specifies which pairs of images are equivalent, and the construction in section 3.5 to generate consistent results.

3.5 Constructing the Topological Map

Finally the decision matrix D is used to define nodes V and determine the map connectivity E_T. D can be viewed as an adjacency matrix of an undirected graph. Since there is no guarantee that D found through belief propagation is symmetric, it is initially treated as an adjacency matrix for a directed graph, and then the direction from all the edges are removed resulting in a symmetric graph $D' = D \vee D^{\mathrm{T}}$. It is possible to use the graph defined by D' as a topological map. However this representation is practically useless because multiple nodes (i.e. images) represent the same location. To achieve compactness, D' needs to be pruned while remaining faithful to the overall structure of the environment. Booij et al. [2005] achieve this by approximating for the minimum *connected* dominating set. By using the temporal assumption, connectedness is no longer a requirement and minimum dominating set can be used to prune D'. The nodes V are found by finding the minimum dominating set of D'. Finding the optimal solution is NP-Complete, however algorithm 3.1 provides a greedy approximation. This approximation has a guaranteed bound of $H(d_{max})$ (harmonic function of the maximal degree in the graph d_{max}) [Chvatal, 1979].

The dominating set K serves as the basis for the final set of map nodes V. Each dominating node $k \in K$ is also associated with the set of nodes it dominates N_k. Each set N_k represents images which have the "same location" with image k acting as a representative key-frame. The sets $N = \{N_k : k \in K\}$ in conjunction with the temporal assumption are used to connect the map T. An edge (k, j) is added if N_k and N_l contain two consecutive images from the sequence, i.e. $(k, j) \in E_T$ if $\exists i$ such that $i \in N_k$ and $i + 1 \in N_l$. This yields our final topological map $T = (N, E_T)$.

Require: Adjacency matrix D'
  $K \leftarrow \emptyset$
  while D' is not empty **do**
    $k \leftarrow$ node with largest degree
    $K \leftarrow K \cup \{k\}$
    $N_k \leftarrow \{k\} \cup Nb(k)$
    Remove all nodes N_k from matrix D'
  end while
  return $K, N = \{N_k : k \in K\}$

Algorithm 3.1: Approximate Minimum Dominating Set

3.6 Experiments

Topological maps were constructed using the above method on five different image sequences. For each sequence, the pair-wise image similarity score matrix M is calculated as per section 3.3. The matrix M is used to detect loop-closures with loopy-belief propagation as described in section 3.4 to generate the matrix D. Finally, D is used to generate the final topological map with approximate minimum dominating as in section 3.5. In addition to using dynamic programming based image similarity score M, results are included for FAB-MAP [Cummins and Newman, 2008a] and two different methods of calculating

image similarity scores.

The two alternate scoring methods are, M^{SIFT} where M_{ij}^{SIFT} = number of SIFT matches between image i and image j, and M_{ij}^{SYM} = number of *symmetric* SIFT matches (the intersection of the matches from image i to image j and the matches from j to i).

3.6.1 Image sets

Three image sequences, *indoors*, *Philadelphia*[1] and *Pittsburgh*[2] were captured with a Point Gray Research Ladybug camera. The Ladybug is composed of five wide-angle lens camera arranged in circle around the base and one camera on top facing upwards. The resulting output is a sequence of frames each containing a set of images captured by the six cameras. For the outdoor sequences the camera was mounted on top of a vehicle which was driven around an urban setting, in this case the cities of Philadelphia and Pittsburgh. In the indoor sequence, the camera was mounted on a tripod set on a cart and moved inside the building covering the ground and 1st floors. Ladybug images were processed independently for each camera using the SIFT detector and extractor provided in the VLFeat toolbox [Vedaldi and Fulkerson, 2008]. The resulting features for every camera were merged into a single set and sorted by their spherical coordinates. The two remaining sequences, *City Center* and *New College* were captured in an outdoor setting by Cummins and Newman [2008b] from a limited field of view camera mounted on a mobile robot. table 3.1 summarizes some basic properties of the sequences used here. All the outdoor sequences were provided with GPS location of the vehicle / robot. For Philadelphia and Pittsburgh, these were used to generate ground truth decision matrices using a threshold of 10 meters. Ground truth matrices were provided for New College and City Center. For the indoor sequence the position of the camera was manually determined using building schematics at an arbitrary scale. A ground truth decision matrix was generated using a manually determined threshold. The entire system was implemented in Matlab with the exception of

[1] Tardif et al. [2008].

[2] The Pittsburgh dataset has been provided by Google for research purposes.

Data Set	Length	No. of frames	Camera Type	Format
Indoors	Not avail.	852	spherical	raw Ladybug stream file
Philadelphia	2.5km	1,266	spherical	raw Ladybug stream file
Pittsburgh	12.5km	1,256	spherical	rectified images
New College	1.9km	1,073	limited FOV	standard images
City Center	2km	1,237	limited FOV	standard images

Table 3.1: Summary of image sequences processed.

the SIFT detector and extractor implemented by [Vedaldi and Fulkerson, 2008].

3.6.2 Parameters

Both the image similarity scores and the MRF contain a number of parameters that need to be set. When calculating the image similarity score, there are five parameters. The first t_{match} is the threshold on th L_1 norm at which two SIFT features are considered matched. In addition, dynamic programming requires three parameters to define the score of an optimal alignment: s_{match}, s_{gap}, s_{miss}. s_{match} is the value by which the score of an alignment is improved by including correctly matched pairs of features. s_{gap} is the cost of ignoring a feature in the optimal alignment (insertion and deletion), and s_{miss} is the cost of including incorrectly matched pairs (substitution). The following values were used: $t_{match} = 1000$, $s_{match} = 1$, $s_{gap} = -0.1$ and $s_{miss} = 0$. Finally $w = 30$ is the window size, used to avoid calculating similarity scores for images taken within very short time of each other. Constructing the MRF requires three parameters, F, σ and α. The normalization factor, F, has already been defined as $\max(M)$ used to re-scale the data in the interval $[0, 1]$. The σ used in defining edge potentials is $\sigma = 0.05F$. Finally $\alpha = 2$ to re-scale the Gaussian to favor edges between similarly valued nodes. Inference using loopy belief propagation features two parameters, a dampening factor $\lambda = 0.5$ used to mitigate the effect of cyclical inference and $n = 20$, the number of iterations over which

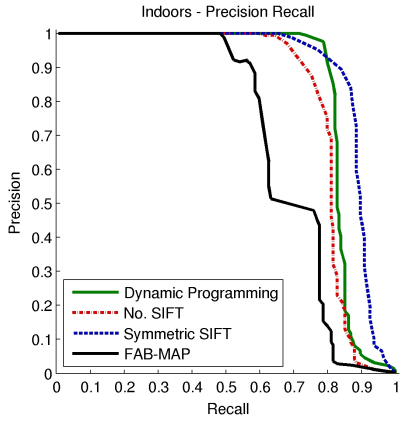
to perform inference.

To process spherical images using FAB-MAP only images captured by camera 0 (Directly forwards / backwards) were used, as these yielded the best results. Recall that FAB-MAP takes into account pair-wise correlations between features, and downplays the contribution of correlated visual words. By using the full spherical image, more features appear in each frame at the same time, increasing correlations between visual words. FAB-MAP also uses the prevalence of visual words to weigh the importance of their contribution. By using spherical images the same features are visible for more frames as they move from front to back along the viewing sphere. This increases the percentage of frames in which a visual word is visible, reducing its contribution.

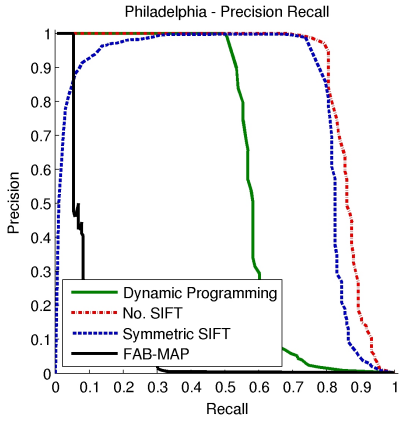
3.6.3 Results

For each sequence figure 3.3 shows precision-recall curves calculated by directly thresholding the image similarity scores. The proposed image similarity measure outperforms state of the art in terms of precision and recall in all sequences. The gain is most pronounced in the *Philadelphia* sequence, where FAB-MAP yields extremely low recall rates. The impact of Loopy-BP inferencing is shown in figure 3.4. Here, for each similarity score, precision-recall is shown compared across different sequences with and without inferencing. Instead of standard precision-recall, interpolated precision was used due to the sparsity of the precision-recall curve after inferencing. Recall that inferencing served two purposes: discretization and temporal smoothness. This greatly reduces the ambiguity in the score matrix, pushing values further towards either extreme, 1 for loop-closure, 0 for not, leaving fewer points. Therefore these plots show interpolated precision-recall. This is defined in eqn. 2 [Everingham et al., 2010] as:

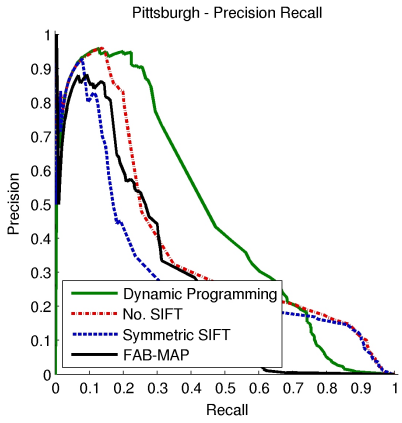
$$p_{interp}(r) = \max_{\tilde{r}:\tilde{r}\geq r} p(\tilde{r}),$$



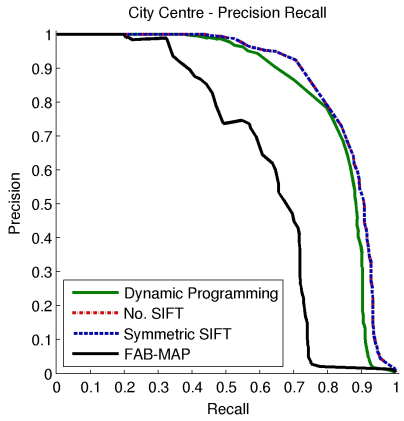
(a) Indoors



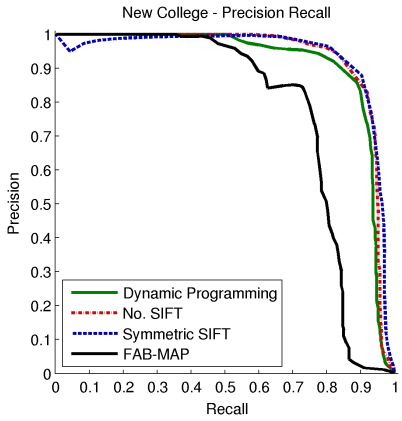
(b) Philadelphia



(c) Pittsburgh



(d) City Center



(e) New College

Figure 3.3: Precision-recall curves for different thresholds on image similarity scores.

	Indoors	Philadelphia	Pittsburgh	City Center	New College
Precision	91.67%	91.72%	63.85%	97.42%	91.57%
Recall	79.31%	51.46%	54.60%	40.04%	84.35%

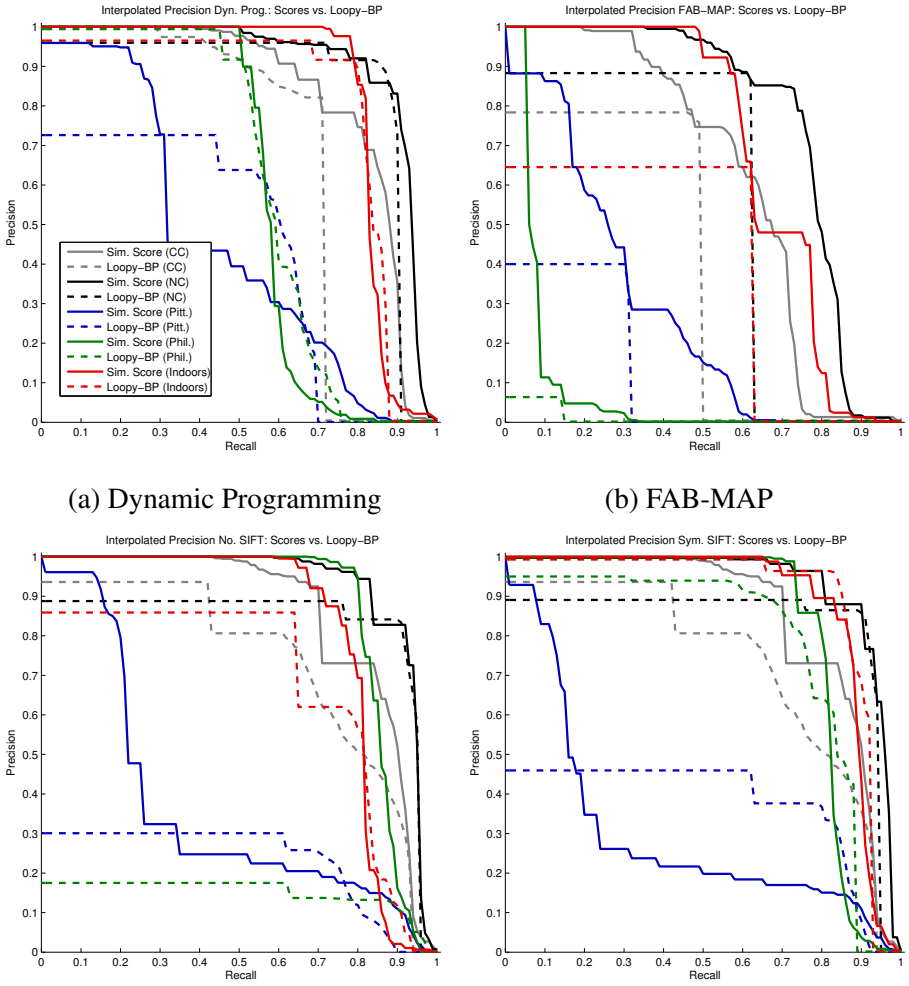
Table 3.2: Precision and recall after performing inference.

for $r \in \{0, 0.01, \ldots, 1\}$. Figure 3.4a shows how inferencing generally improves precision-recall for all dynamic programming scores. In the case of the FAB-MAP image similarity measure (figure 3.4b), inferencing provides no benefit, simply reducing the available ranges of precision and recall by reducing the number of distinct values. The reason is that FAB-MAP scores are normalized and have already undergone smoothing, and so undergo additional smoothing. On the other hand, figure 3.4c and figure 3.4d show little gain from the inference step for the remaining similarity scores (No. of SIFT matches and symmetric SIFT matches). Since these scores do not contain geometric verification, the inference step is more likely enhances incorrect matches reducing precision. The symmetry mitigates this and so is less affected by errors due to smoothing in the inference. The final precision-recall values for proposed approach of cyclically invariant dynamic prgramming with inferencing are shown in table 3.2.

Finally in figure 3.5 the topological maps resulting from running dominating sets on the decision matrices D are displayed. The ground truth GPS positions are used for display purposes only. The blue dots represent the locations of the key-frames K with the edges E_T drawn in blue. Red dots mark key-frames which are also loop-closures. For reference, figure 3.6 provides ground truth maps and loop-closures.

3.6.4 Failure Cases

Although the system is designed to account for both perceptual aliasing, and perceptual variability, they are still a problem. In the case of empty scenery (e.g. highways, large parks), the images do not contain enough detail to form discriminative feature sequences.



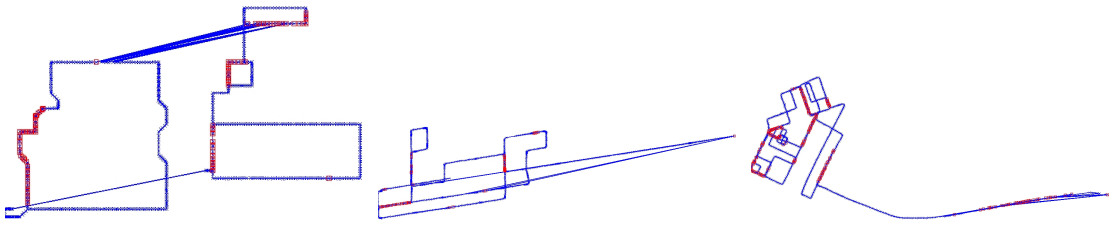
(a) Dynamic Programming

(b) FAB-MAP

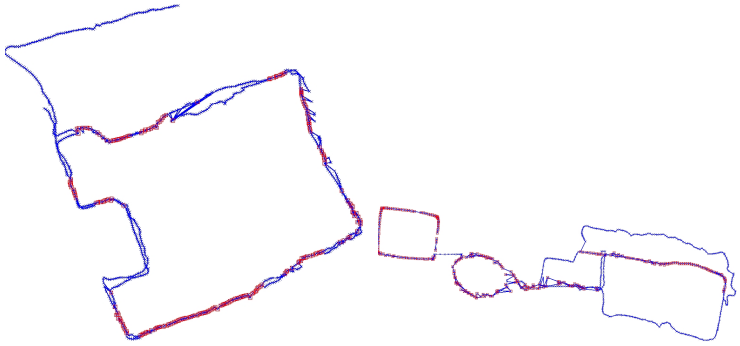
(c) No. of SIFT

(d) Symmetric SIFT

Figure 3.4: Interpolated precision-recall plots for different image similarity measures. The original scores (solid lines) compared with the same score after Loopy-BP inference (dashed) across all image sequences. These are City-Center (gray), New College (black), Pittsburgh (blue), Philadelphia(green) and Indoors (red)

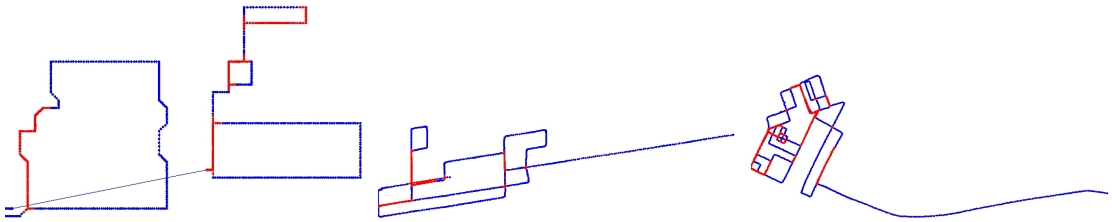


(a) Indoors (b) Philadelphia (c) Pittsburgh

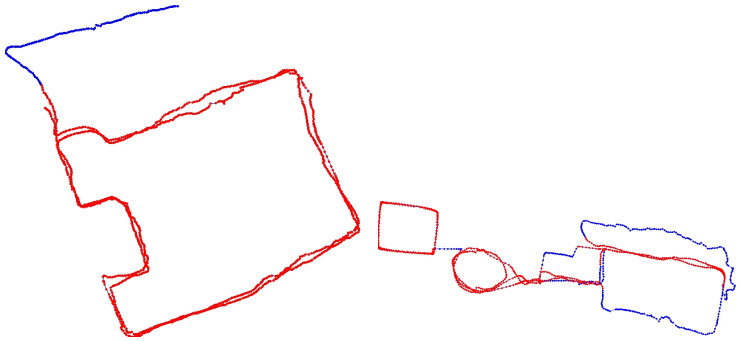


(d) City Center (e) New College

Figure 3.5: Loop-closures generated using minimum dominating set approximation. Blue dots represent positions of key-frames K with edges E_T drawn in blue. Red dots mark key-frames with loop-closures.



(a) Indoors (b) Philadelphia (c) Pittsburgh



(d) City Center (e) New College

Figure 3.6: Ground truth maps and loop-closures. Blue dots represent positions of key-frames K with edges E_T drawn in blue. Red dots mark key-frames with loop-closures.

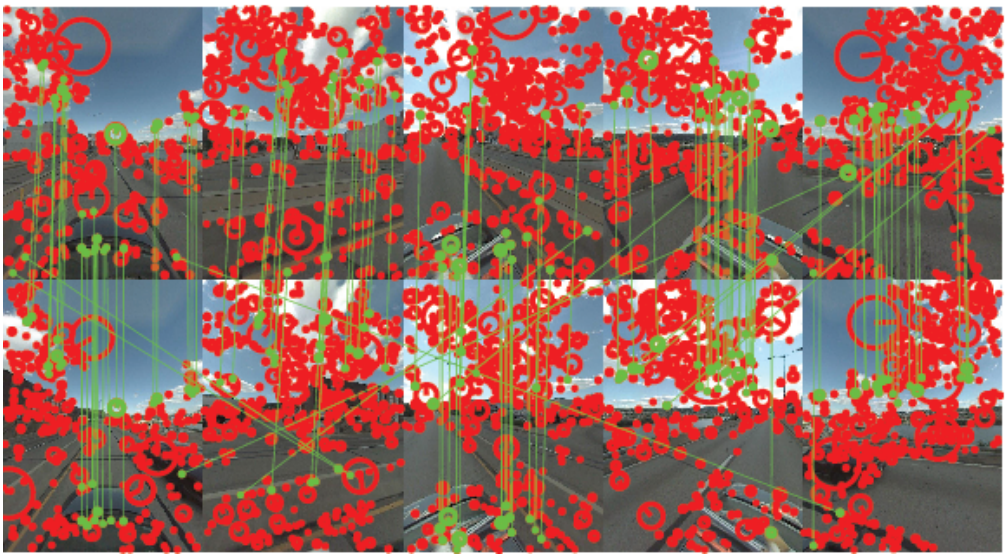
Here perceptual aliasing is two-fold: lack of distinguishing features, and repeated detection of irrelevant features which are visible from different locations such as faraway landmarks. An example of an open highway at the tail end of the Pittsburgh data exemplifies this. figure 3.7a and figure 3.7b were taken more than 400 meters apart, but are visually very similar. The detected SIFT features, and their matches are displayed in figure 3.7c. Unmatched features are red, while the features marked in green are those matched across the images thus raising the image-similarity score. These include both non-discriminative features seen in every image such as the road, and the vehicle, and faraway landmarks such as cloud formations which are visible from many locations.

On the flip side, repeated identical structures are also very challenging. These can manifest both outdoors (e.g. housing developments and planned communities), and indoors with shared floor-plans and common interior design (e.g. apartment complexes and office buildings). figure 3.8a and figure 3.8b were taken at the same indoor location, but on different floors. The detected SIFT features, and their matches are displayed in figure 3.8c. Unmatched features are red, while the features marked in green are those matched across the images thus raising the image-similarity score.

Both of these examples would prove be difficult to distinguish even for a human being. In the case of the highway, there is little that can be done to distinguish between these locations. It is possible to mask out the vehicle and discount those features reducing the overall match score, but this does not add distinction between the scenes. However, in the case of the indoor sequence, incorporating additional information could potentially improve precision, distinguishing between the two scenes. This could be accomplished by incorporating textual information from signage (names, room nos.) or by detecting changes in elevation (by detecting ramps or stairs) and direction.

(a)

(b)

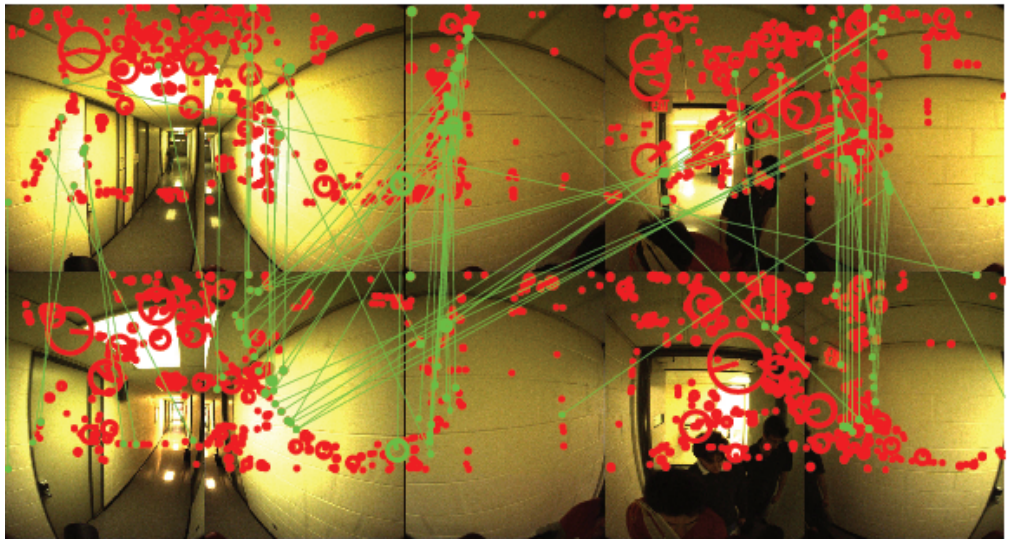


(c) SIFT features with matched pairs in green

Figure 3.7: Examples of a false-positive loop-closure due to lack of visual information and perceptual aliasing. Two images captured on an open highway taken more than 400 meters apart, (a) and (b). The same images overlayed with detected SIFT features, with matching features in green, and unmatched features in red (c).

(a)

(b)



(c) SIFT features with matched pairs in green

Figure 3.8: Examples of a false-positive loop-closure due to visually identical locations. Two images captured in the same building, and location but different floors, (a) and (b). The same images overlayed with detected SIFT features, with matching features in green, and unmatched features in red (c).

3.7 Summary

In this chapter the challenge of building topological maps in urban settings was addressed. Purely topological maps are constructed from video sequences captured from moving vehicles in urban settings and from a moving platform in an indoor setting. Since only the videos are used, the proposed system is purely visual. The underlying assumption is that the images are presented in a temporally consistent manner. A highly accurate image similarity score is found by a cyclical alignment of sorted feature sequences. This score incorporates weak geometric queues of visual features in a spherical image instead of relying traditional approaches which use expensive epipolar geometry for verification. To verify feature geometry, sequences of features are aligned using a rotationally invariant version of dynamic programming. By combining both appearance and layout the challenge of perceptual aliasing is confronted, resulting in fewer incorrect loop-closures. This score is then refined via Loopy-Belief Propagation to make a final decision to the existence of loop-closures. The image similarity scores are used as priors for a Markov Random Field lattice, which is used to find temporally consistent loop-closures. Integrating information across multiple frames over time increases the accuracy of the initial detections, and is better equipped to address both perceptual aliasing and perceptual variability. Finally the images are clustered into nodes using the recovered loop-closures and approximate minimum dominating set. This yields a compact representation of the environment. The semantics in the system are derived from the temporal assumption in the loopy-belief propagation and the temporal component in the minimum dominating set algorithm, and the bearing-only simplification used for geometric verification. The resulting topological map can be used for either path planning or for bundle adjustment in visual SLAM systems.

The issue of scalability has yet to be addressed, and little emphasis has been placed on improving speed. The bottleneck of the system is computing the image similarity score. In some instances, taking over 166 hours to process a single sequence while FAB-MAP [Cummins and Newman, 2008a] accomplishes the same task in 20 minutes. The speed of the system can be improved in several ways. In addition to implementing score

calculation with a parallel algorithm (either on a multi-core machine or using graphics hardware), it may be possible to construct approximations to the devised image similarity score. These include using visual bags of words in a hierarchical fashion [Nister and Stewenius, 2006] and building the score matrix M incrementally [Valgren et al., 2006, 2007]. Such approximation can also provide bounds on the score, allowing the use of early "bail-out" [Cummins and Newman, 2010a] or branch-and-bound algorithms. An additional pre-filtering step using either a global feature such as GIST [Oliva and Torralba, 2001] or robot orientation [Liu et al., 2010], is another possibility.

In addition to improvements in scalability, the addition of other features have the potential to add robustness to perceptual aliasing and variability. Incorporating additional semantic information in the form of text recognition, and robot orientation [Liu et al., 2010] could help distinguish between visually similar locations or better recognize previously visited locations.

In order to improve recognition under changes in lighting, perspective and field of view, something more robust than local image features can be used. One such possibility is to use an object detector to recognize objects seen at different times in the same location. The next chapter uses object detections as a more robust feature in an effort to deal with the challenge of perceptual variability to solve the problem of robot localization.

Chapter 4

Robot Localization with Soft Object Detection

4.1 Introduction

The previous chapter dealt with the large scale problem of creating a topological map of an environment from a video sequence, mainly urban environments. In this chapter, the focus shifts to a smaller scale, the problem of localization within an known environment, including a large open space. Research on SLAM, metric or topological, has flourished in the past decade producing maps in terms of point clouds, occupancy grids, or graphs of poses and landmarks. Such maps are used by robots to localize themselves and re-traverse the mapped space. While the probabilistic inference used here is the same as in the classic probabilistic map-based localization [Dellaert et al., 1999], the underlying representation is completely different for both the prior map and the sensorial data.

Traditionally, vision-based localization is achieved by matching local image features (e.g. SIFT, SURF,) computed and detected in images. The process of mapping these features is expensive, generally requiring full traversal of the environment with a complete 3D SLAM system. On top of that, the underlying representation is complex, consisting of hundreds or thousands of highly localized points scattered throughout the environment.

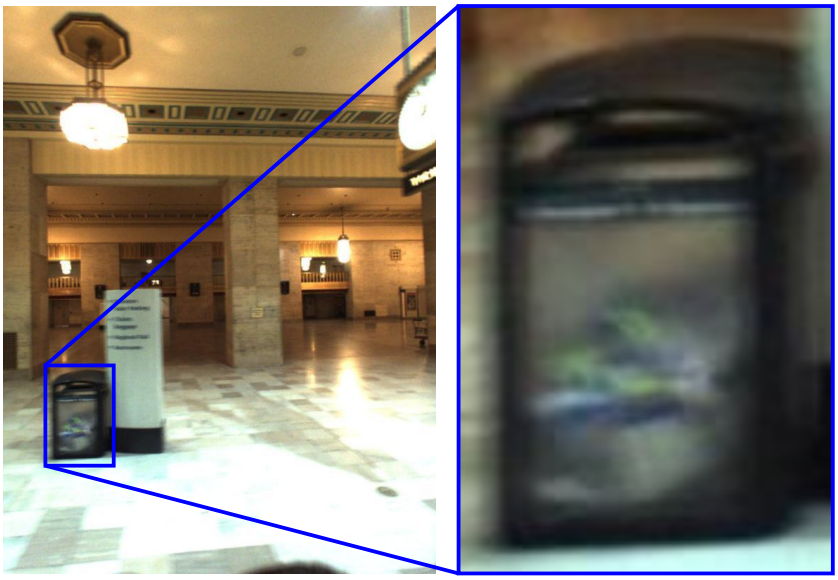
Transfering this map, and the knowledge of these points is non-trivial. Finally, the process of matching these features is vulnerable to changes in viewpoint, illumination variations and differences in camera hardware.

In contrast, there is an abundance of prior maps for GPS-denied environments which are semantically annotated, some as simple as a rough sketch. For many of these semantic annotations represent the positions on the map of objects of known classes. By using objects in lieu of local features localization becomes more robust to changes in viewpoint and illumination variations. Obviously, these advantages are offset by the increased complexity required to learn and detect objects. Furthermore, traditional object detectors are ill-suited for this task as they are heavily biased towards images in which the object is centered and well focused. Although these assumptions are safe in a traditional object detection task, they are not true in a localization setting. Not only do images from a moving camera suffer from bad focus and motion blur, but the desired objects are generally found off-center, away from the camera's motion path (See figure 4.1).

Therefore object-based localization poses several distinct challenges. As in the previous chapter, perceptual aliasing needs to be accounted for. In this scenario perceptual aliasing takes the form of object layout, as opposed to visual similarity of low-level features. Two locations with similar objects in similar configurations could get mistaken for each other. However, perceptual variability is essentially gone from localization, and is instead relegated to object detection directly. As long as the map of the environment is up-to-date, the signature of a location remains stable and localization is immune to perceptual variability. Instead, failures in object detection can incorrectly cause variability in the perceived composition of objects at a location and lead to incorrect localization. Therefore using objects to localize in an environment separates perceptual variability from localization. To these is added the difficulty of performing object detection itself under the conditions mentioned earlier.

In order to tackle object detection in this setting the proposed approach departs from the traditional object recognition and localization paradigm by avoiding hard detection of

(a)



(b)

Figure 4.1: Typical images from a moving platform. (top) Frontal view, object is off-center with severe perspective distortion. (bottom) Lateral view, significant motion blur.

objects. Instead, a *heatmap* is produced from the image for every object, representing the probability that the object is present at a particular scale and at a particular position. Such heatmaps might be multi-modal, indicating the existence of multiple objects or the *hallucination* of multiple objects (i.e. false positives). Given several object classes, the image is then represented with a vector at each pixel containing all detection scores for that pixel. This is equivalent to a projection of the image on a basis consisting of templates representing the object classes. In addition to addressing the difficult problem of object detection in a non-ideal setting, this also confronts the problem of perceptual variability. Since no threshold is applied, weak signals are not discarded, leaving a more robust and dense representation.

Using a labeled map of the environment the soft detection signature (or heatmap) of a query image is filtered using expected presence of objects. This response is marginalized over objects, and along vertical lines to yield a bearing only representation. The score vector over bearings is combined with prior locations of objects to estimate 2D position and orientation using particle-filter based localization. By integrating responses over time, the system combines information from multiple locations, reducing the ambiguity and the effect of perceptual aliasing. Experiments are included for two types of scenarios. First, a very large indoor space (urban train station) with a hand annotated prior map. The second scenario is composed of a set of video sequences in office environments with objects laid out on a surface. Results on the first dataset show that short sequences of panoramic images are sufficient for localization using only a few object categories. Experimental results on the second dataset show localization using generated maps.

To summarize, the following contributions advance the state of the art:

- Solving the localization problem using prior maps containing objects instead of point clouds or visual features.

- A new image representation that instead of containing signal to symbol translation[1],

[1] In standard object detection the image (i.e. the signal) is replaced by a list of labeled bounding boxes (i.e. symbols).

contains at every pixel a signature of all object-detection scores avoiding a hard-detection commitment. Precision-recall curves are therefore not necessary because no threshold is applied.

- Object detection scores can be established with very simple representations, such as color or gradient distributions. Alternatively they can be extracted from traditional object detectors. Both are sufficient to localize the robot with a particle filter.

- A novel likelihood formulation for modelling objects in a scene using a dense signature of object-detection scores.

The structure of the chapter is as follows. Section 4.2 reviews the related work. Section 4.3 presents the first soft object-detection strategy while section 4.4 described the particle filter based localization. These are validated with the first set of experiments in section 4.5. The model is extended in section 4.6 with the second object-detection strategy and section 4.7 with the novel observation likelihood formulation. Section 4.8 includes experiments to validate those. Finally section 4.9 summarizes the chapter.

4.2 Related Work

Using semantic information for localization has been motivated by Kuipers [2000] spatial semantic hierarchy paradigm. Ranganathan and Dellaert [2007], Atanasov et al. [2014, 2016] and Salas-Moreno et al. [2013] all employ object detection in an effort to localize the camera. Ranganathan and Dellaert [2007] use a generative model for a place which has the form of a 3D constellation with object attributes of shape and appearance. While the model is probabilistic, the object detection produces a "hard" uni-modal distribution as opposed to a "soft" detection modeling the probability of having an object at each bearing as presented here. Atanasov et al. [2014, 2016] on the other hand do not define a "place", but rely instead on a global object map (similar to this chapter). However, they also rely on "hard" detections and use the matrix permanent to solve the data association problem

between detections, and mapped instances. Finally, SLAM++ [Salas-Moreno et al., 2013] use 3D meshes to detect objects in a scene and localize relative to these. By capturing dense 3D information at every frame, the system is able to both localize w.r.t to objects and map them. Localization is maintained by continuously tracking objects. Tracking failures result in the creation of a new independent local map which is discarded once re-localization is successful.

Soft detection has been applied by Li et al. [2010a] in "object bank" response maps, using a large number of pre-trained generic object detectors with the goal of scene classification. This works has also been inspired by the concept of *classemes* introduced by Torresani et al. [2010] for novel category discovery while here it is used for location modeling.

Most of the research in object-based localization is tailored for small indoor environments like offices with simple topology and few object categories. Many approaches detect doors or gateways [Murillo et al., 2008; Rituerto et al., 2012; Schroter et al., 2002; Stoeter et al., 2000] for place recognition as well as for detection of passages in a topological sense. Espinace et al. [2010] detect the semantics of spaces (kitchens, etc.) and the objects therein starting from metric and topological maps in an indoor environment. They use both appearance features as well as 3D geometry to detect seven objects in four scene categories. Galindo et al. [2005] apply a conceptual hierarchy of things, objects, and rooms to label existing maps. Vasudevan et al. [2007] detect objects as well as passages in order to categorize and recognize places. The notion of location as presented here is similar to cluster in the work of Posner et al. [2008] although it uses directly low level features and not objects.

Several approaches can be used to automatically produce the prior semantic maps expected in this chapter. Given dense 3D point clouds, both SLAM++ [Salas-Moreno et al., 2013] and Wong et al. [2013, 2015] can be used to generate object maps. Civera et al. [2011] apply appearance and geometry based recognition to annotate feature maps established with monocular SLAM. Similar annotation of features or regions on top of SLAM

are undertaken in Oberlander et al. [2008]; Trevor et al. [2010]. Wolf and Sukhatme [2008] label 3D maps based on traversability of terrain using hidden-Markov-model and support-vector-machine techniques. A different approach producing maps consisting only of semantic entities (relational object maps) has been introduced in [Limketkai et al., 2005] by modeling spaces with relational Markov networks. Finally, chapter 5 presents a framework for creating 3D semantic maps of objects in small-scale 3D environments.

4.3 Object Detection

Traditional object detectors yield one of the following representations when processing a query image: boolean flag [Lazebnik et al., 2006], bounding boxes [Felzenszwalb et al., 2010], and full object (or clutter) segmentation [Toshev et al., 2010]. These often include a score (or probability) measure of the detection as a whole, but always culminate in a hard decision (i.e., indicated by a bounding box) as to the presence of an object.

Instead of relying on bounding boxes or image segmentation, object heatmaps are used instead. These are confidence maps of the the object being present. They provide a value for each pixel (or block of pixels) and give a dense-detection response for a given query image, as opposed to a sparse-detection response. They are normalized in a global fashion to ensure that results for different templates and different features are comparable.

Heatmaps are computed for two types of local image features. These are histograms of *normalized* gradient energies (HOGE) computed over blocks of pixels (section 4.3.1) and histograms of quantized colors (HQC, section 4.3.2). Matching is performed separately with each feature. They are normalized in a global fashion to ensure that results for different templates and different features are comparable. Then the resulting heatmaps are multiplied together to yield the final object detection heatmap (section 4.3.3).

Any number of additional image features can be combined with this approach. There are only two properties required for an image feature to be used: first, that it is computed densely over the image, yielding a detection value for every pixel (or block of pixels);

second, to successfully combine the output of an additional feature, it must be possible to normalize its detection results independently of the template size and the support-region size. These requirements are very flexible and, therefore, the system can incorporate a variety of additional image features to increase discriminability and improve results.

4.3.1 Histogram of Gradient Energies (HOGE)

The desired spatial orientation measurements are realized via filtering using a set of Gaussian derivative filters, point-wise squaring and summation over a given spatial region,

$$E_{\hat{\theta}}(u,v) = \sum_u \sum_v \Omega(u,v)[G_{N_{\hat{\theta}}}(u,v) * I(u,v)]^2, \tag{4.1}$$

where $I(u,v)$ denotes the input image, $*$ convolution, $\Omega(u,v)$ a mask defining the integration region, and $G_{N_{\hat{\theta}}}(u,v)$ the Nth derivative of the Gaussian with $\hat{\theta}$ the filter's orientation.

The initial definition of local energy measurements (equation (4.1)), is confounded by local image contrast. This makes it indeterminate whether a high response in the filtered imagery, indicates the presence of the particular spatial orientation or instead is a low match but yields a high response due to strong image contrast. To remove contrast-related information, the energy measures, are normalized locally by the ensemble of oriented responses at each point,

$$\hat{E}_{\hat{\theta}_i} = \frac{E_{\hat{\theta}_i}}{\epsilon + \sum_{\hat{\theta} \in \mathbb{S}} E_{\hat{\theta}}}, \tag{4.2}$$

where $\mathbb{S}$ denotes the set of considered oriented energies, (equation (4.1)), and ϵ is a constant that serves as a noise floor (set to 1% of the expected maximum filter response). In addition, a normalized ϵ is computed, as in (equation (4.2)), to explicitly capture lack of structure within the region delineated by $\Omega(u,v)$ [2]. The result is a distribution (i.e. histogram) within a given region of support, $\Omega(u,v)$, indicating the relative presence of a particular set of spatial orientations within neighborhoods of the input imagery. Finally,

[2]Note that regions where structure is less apparent, e.g. , region of texture-less wall, the summation in the denominator approaches zero; hence, the normalized ϵ approaches one and thereby indicates lack of structure.

to define the template representation, the image is divided into non-overlapping regions, $\Omega_i(u, v)$, and a normalized energy histogram is computed for each region (see Fig. figure 4.2b).

In summary, equations (4.1) to (4.2) culminate in a distribution (histogram) indicating the relative presence of a particular set of spatial orientations within neighborhoods of the input imagery. Significantly, the derived measurements are invariant to additive and multiplicative bias in the image signal, due to the band-pass nature of (equation (4.1)) and the normalization (equation (4.2)), respectively. Invariance to such biases provides a degree of robustness to various potentially distracting photometric effects (e.g., overall scene illumination, sensor sensitivity). Owing to the oriented energies being defined over a spatial support region (equation (4.1)), the representation can deal with input data that are not exactly spatially aligned. Owing to the distributed nature of the representation, clutter can be accommodated: Both the desirable pattern structure and the undesirable clutter-related structure can be captured jointly so that the desirable components remain available for matching. Finally, the representation is efficiently realized via linear (separable convolution, point-wise addition) and point-wise non-linear (squaring, division) operations; thus, efficient computations are realized [Freeman and Adelson, 1991].

4.3.2 Histograms of Quantized Colors (HQC)

To incorporate color information, color histograms of images are computed. The RGB color model is considered here, but different color models can also be considered. An image is quantized from RGB into an indexed color space. The target color map is created by uniformly sampling the RGB cube in all three channels. Quantizing each channel into $k << 256$ bins generates a colormap with k^3 distinct RGB values (here $k = 4$). Each pixel in the image is then mapped to the closest value in the target colormap. Once quantized, a histogram of the color indices is computed for each block of $n \times n$ pixels. Finally each histogram is normalized to unit energy (sum of values equal 1) by dividing each histogram by the number of pixels per block.

Although this representation is not invariant to large changes in illumination, by drastically reducing the size of the color space, in this case to k^3. it eliminates small changes in illumination lost in the quantization (figure 4.2c).

4.3.3 Matching

The output of the matching step is not a list of bounding boxes, but rather a two dimensional heatmap. A heatmap for each object category is computed via the Bhattacharyya similarity measure [Bhattacharyya, 1943] of the object template features and query features. The maximum responses over all scales is selected and results from different features are multiplied together to yield the final heatmap.

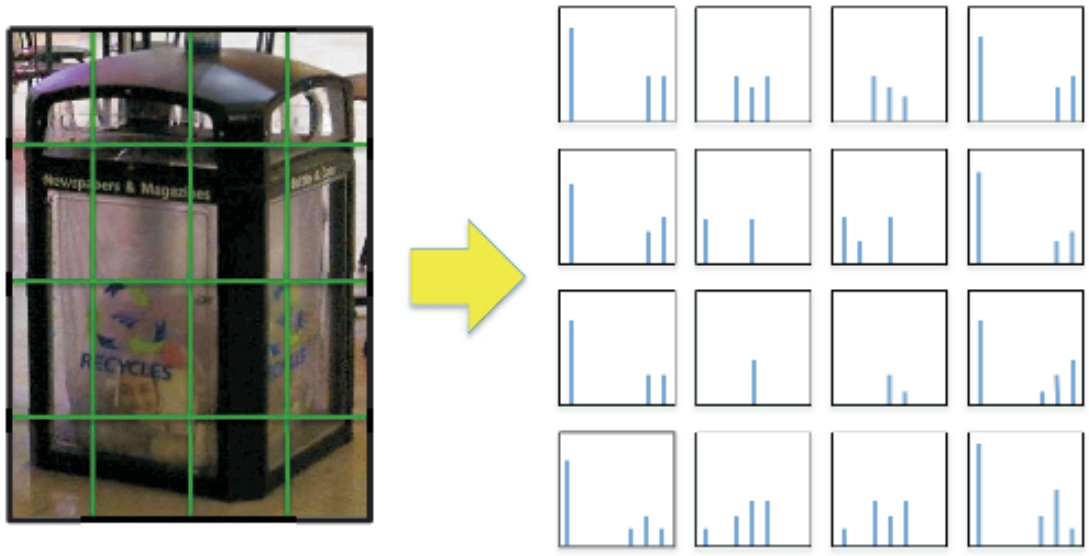
Formally, for a $m \times n$ image I, the heatmap for a specific object template T maps every pixel coordinate (u, v) to a value in $[0...1]$:

$$H_T^F(I) = \max_{Scales} \frac{Corr\left(\sqrt{F(I^s)}, \sqrt{F(T)}\right)}{b_T}, \tag{4.3}$$

where F is the image feature function (either HOGE, or HQC), and I^s represents the image at scale s. $Corr$ is the standard correlation function and b_T is the number of blocks in the template. Dividing the result by the template size scales the values to the range $[0...1]$. The heatmaps from each type image feature are combined using point-wise product:

$$H_T = H_T^{HOGE}(I) \cdot H_T^{HQC}(I). \tag{4.4}$$

Example heatmaps for HOGE, HQC, and their product are shown in figure 4.3. By computing these features densely over blocks of pixels, the resulting template is able to handle small changes in translation, rotation and focus when processing a query image including instances of motion blur or bad focus. Quantizing the colors of the template provides some invariance to illumination. Together these aid in overcoming perceptual variability between different instances of the same object when viewed from different angles, and different instances of the same object category when viewed at new locations. Furthermore, by using a dense template representation of the object, as opposed to a sparse set of feature



(a)

(b) (c)

Figure 4.2: Object Feature Computation. (a) Feature histograms over uniform pixel blocks (b) Histogram of gradient energies with 8 orientations (c) Quantized to 64 color image.

points, the detector is more robust to perceptual aliasing. A more complete representation of the object makes it less likely that a false match will yield a high score.

4.4 Object-Based Localization

The goal of localization is to retrieve the robot's absolute position in the environment using the available information from its on-board sensors, such as wheel encoders and cameras. From computer vision, it is known that the absolute position of a single calibrated camera can be inferred from a minimum of three 3D-2D correspondences, that is, the 3D absolute positions of three scene points and the 2D coordinates of their projections in the camera image [Gao et al., 2003; Kneip et al., 2011]. This method is known as the "perspective three-point algorithm" (P3P). There are three drawbacks in using P3P for object-based localization. First, by using objects in lieu of points, the positions in the image are not as well localized as with points, affecting the accuracy of the triangulation. Even with bounding boxes, the location of the object is imprecise and is affected by foreshortening and occlusions. This is in contrast to feature based approaches which give position information at the pixel resolution. The second difficulty comes from using a soft detector, where only a confidence (i.e. heatmap) that the object is at a given image coordinate. By not committing to a specific object position, the location of the object in the image is not well-defined. This is compensated for by using the whole heatmap as the representation, yielding a more robust detector, but again disallows methods which rely on exact positioning. Finally, P3P requires that three objects be viewed by the robot simultaneously, a situation that cannot be guaranteed in real environments. Therefore to accurately localize, information needs to be combined across multiple frames, integrating detection scores over time. Finally, the use of a dense representation coupled with a dense state-space makes modelling the distribution prohibitive. For these reasons, *particle filter localization* [Dellaert et al., 1999; Thrun et al., 2001] is used.

In probabilistic map-based localization, the state of the robot at the current time step

t is estimated given the knowledge about its initial state and all the measurements up to the current state. In this setup, the robot moves in a planar environment, therefore the state vector is $\mathbf{x} = (x, y, \theta)$, with ($x$,$y$) denoting the robot position and θ its heading. The measurement at time t, $z_t = \{z_t^1, \ldots, z_t^k\}$, is the sequence of heatmaps for all k object classes for image I_t. The set $Z_{1:t} = \{z_1, \ldots, z_t\}$ denotes all measurements from time 1 to time t. Additional information is provided by the map $M = \{o_1, \ldots o_n\}$ which is composed of n object instances with their approximate locations, and class labels, $o_i = (x, y, c)$. The particle filter represents the probability distribution $p(\mathbf{x}_t \mid Z_{1:t}, M)$ of the robot pose by a set of N particles $S_t = \{s_t^i, i = 1..N\}$ drawn from it.

4.4.1 Prediction Update

In this phase, a set of particles S_t are computed from the previous set S_{t-1} by sampling from the motion model. The motion model of a differential-drive robot is used:

$$\begin{cases} x_t^i &= x_{t-1}^i + (v_t + \Delta V)\cos\left(\theta_{t-1}^i + \Delta\theta/2\right) \\ y_t^i &= y_{t-1}^i + (v_t + \Delta V)\sin\left(\theta_{t-1}^i + \Delta\theta/2\right) \\ \theta_t^i &= \theta_{t-1}^i + \Delta\theta \end{cases}, \tag{4.5}$$

where $\Delta\theta = (\omega_{t-1}^i + \Delta\Omega)\Delta t$ and v_t and ω_t are the translational and angular control speeds respectively. Finally, ΔV and $\Delta\Omega$ are normally-distributed random variables that account for the noise in the motion.

4.4.2 Perception Update

In the second phase, the information from heatmaps $\mathbf{z}_t = \{H_{T_1}(I_t), \ldots, H_{T_n}(I_t)\}$ is incorporated. $\mathbf{z}_t$ is the collection of heatmaps for all object categories $\{T_1, \ldots, T_n\}$ for the current image I_t. Each sample in S_t is weighted by w_t^i which needs to be high when a particle is more likely to represent the correct location, and low otherwise. Therefore w_t^i should be defined in a manner proportional to the likelihood of a location given heatmaps: $w_t^i \propto p(\mathbf{s}_t^i|\mathbf{z}_t)$, so that it is representative of the likelihood of $\mathbf{s}_t^i$ given $\mathbf{z}_t$. Finally, the

new set S'_t is computed from the weighted set using *importance re-sampling* [Thrun et al., 2001].

The delicate part is therefore how the weight w_t^i is computed (From this point onward, the subscript t will be omitted to simplify the notation). Since the weights are to be proportional to the likelihood $p(\mathbf{s}^i|\mathbf{z})$ they incorporate both the observed heatmaps $\mathbf{z}^i$ and the particle pose $\mathbf{s}^i$. In order to tie together position ($\mathbf{s}^i$) with observation ($\mathbf{z}^i$) a semantically annotated map, M, is used. The map $M = \{m_1, \ldots, m_o\}$ is used to generate the *expected* heatmaps, $\hat{\mathbf{z}}^i$, for each object category at the particle's position. $\hat{\mathbf{z}}^i$ is compared with the *observed* heatmap, $\mathbf{z}^i$, to calculate the particle's weight. A good measure of the similarity of two functions is their inner product; therefore, the following expression is used for w^i:

$$w^i = \sum_{j=1}^{n} < \mathbf{z}_j^i, \hat{\mathbf{z}}_j^i >, \tag{4.6}$$

where the subscript j denotes a specific object category. The weight of each particle is then a sum of the inner products between the observed and the expected heatmap of each object category.

Recall that the heatmap $\mathbf{z}^i$ for object category is a two dimensional function of the image coordinates. The *expected heatmap* $\hat{\mathbf{z}}^i$ is recovered using the particle's position s^i and the map M. However, only planar motion is considered so it is reasonable to convert the heatmap into a one-dimensional signal that depends only on the bearing angle δ_k^i of the object m_k with respect to the particle s_i. Since the camera is calibrated and approximately perpendicular to the ground plane, a 2D-to-1D conversion is performed by simply taking the maximum along each column of the heatmap image. Like the 2D heatmap, the 1D heatmap also assumes values in the range $[0...1]$. An example 1D heatmap is shown in figure 4.3 (bottom) for the case of the clock template in figure 4.4. The expected heatmap $\hat{\mathbf{z}}_j^i$ for a given particle $\mathbf{s}^i$ and for a specific object category j is computed as,

$$\hat{\mathbf{z}}_j^i(\delta) = \max_{m_k \in T_j} \Phi(s_i, m_k) \exp\left(-\frac{(\delta - \delta_k^i)^2}{2\sigma^2}\right), \tag{4.7}$$

with $\delta \in [0 \ldots 2\pi]$ and $\Phi(s_i, m_k)$ is used to disregard objects too far away from the particle

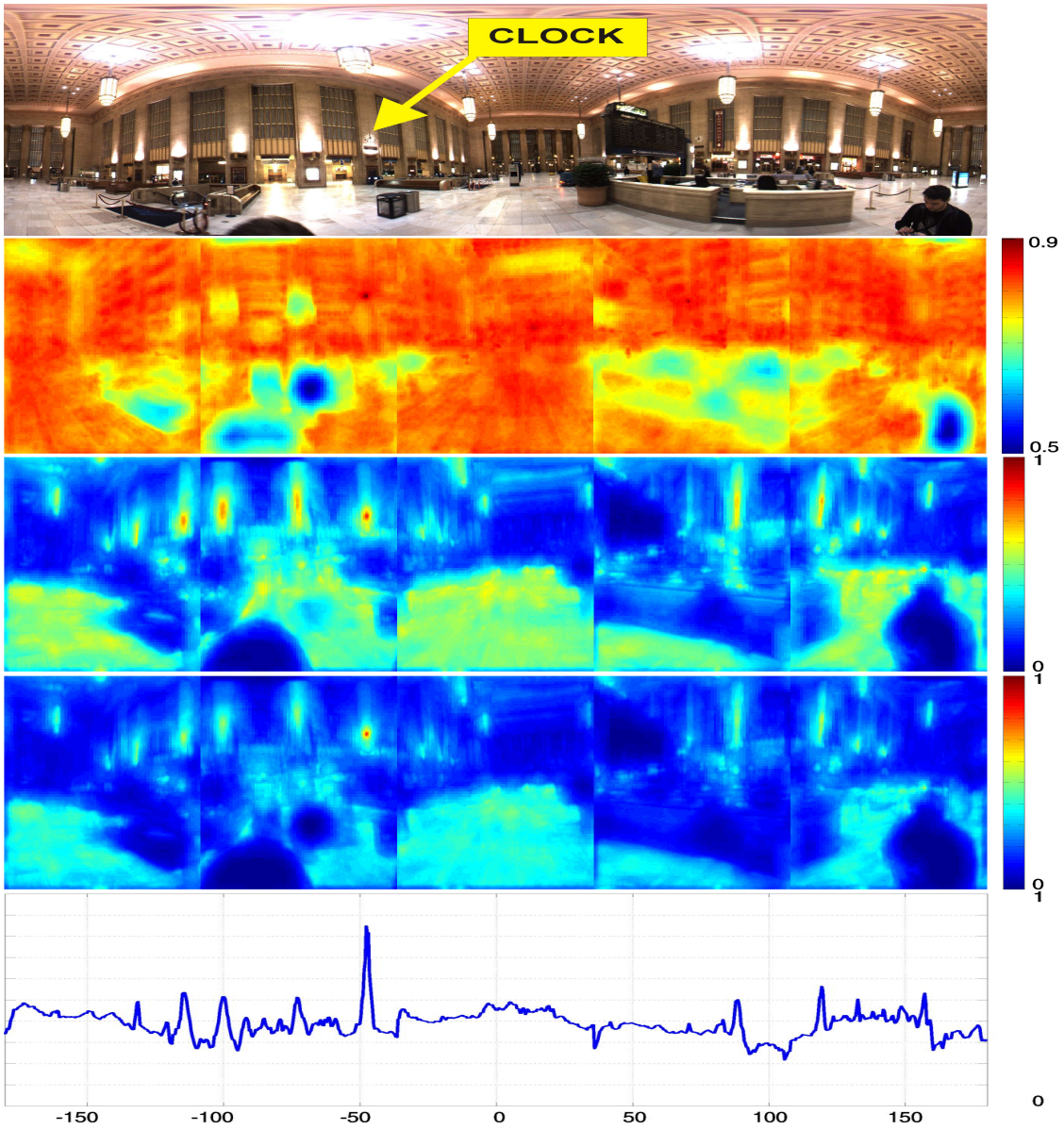


Figure 4.3: Original 360° panoramic image (Top row) followed by the HOGE heatmap, the HQC heatmap. The final heatmap is computed as point-wise product between the HOGE and the HQC heatmaps (4th row). Finally the one-dimensional heatmap (Bottom row). These heatmaps were computed for the clock. Notice the well distinguishable peaks in the heatmaps in correspondence of the clock.

to be visible,

$$\Phi(s_i, m_k) = \begin{cases} 1 \text{ for } ||s_i - m_k||_2 \leq dist_v \\ 0 \text{ for } ||s_i - m_k||_2 > dist_v \end{cases}, \tag{4.8}$$

with $dist_v = 20$. The same σ is used for every object, regardless of its distance to the particle. The $\max(\cdot)$ operator is used instead of the $\sum(\cdot)$ operator so that if two instances are occluding each other their heatmaps do not sum up.

By using the inner product as a heatmap similarity measure, the absence of objects does not impart a negative contribution, but only the lack of a positive one. This could lead to incorrect localization when the negative data pertains to a discriminative object. The importance of ignoring negative data is seen when one considers that observed heatmaps are real signals, while the expected heatmaps are synthetically composed from mixtures of Gaussians (equation (4.7)). If negative data is taken into account (e.g. defining $w_i = \sum_{j=1}^{n} ||\mathbf{z}_j^i - \hat{\mathbf{z}}_j^i||_2$), noise from the observed heatmap will have a detrimental impact on similarity scores. This noise is generated by object template responses to the background of the scene and scene clutter. In traditional object detection, this noise is avoided by performing non-maxima suppression on and thresholding detector results. However, by choosing not to threshold detector responses and using the dense representation instead, this noise is unavoidable. Therefore, using the inner product ensures that weak correct signals are respected when objects are expected, while ignoring template responses when no object is expected.

The last key component of equation (4.7) is the visibility threshold function $\Phi(s_i, m_k)$ which disregards far-away objects. This is in order to avoid polluting the expected heatmap $\hat{z}$ with objects that the detector will not be able to perceive. Recall that the object detector from section 4.3 uses a block representation of the template. As such, the region of interest covered by a distant object will not only shrink due to distance, but will also shrink when filter responses (HOGE) or colors (HQC) are aggregated over blocks of pixels. Therefore, as the object gets more distant, the template gets smaller, and both edge and color information is aggregated over fewer bins, resulting in less discriminative detections.

In addition to the practical consideration of the inability to reliably detect objects from large distances, the benefit for including distant objects is limited. The further away an object is from the camera, the less reliable an indicator for location it becomes (Recall figure 3.7 from the previous chapter). At the extreme case, very far objects, or objects at infinity (e.g. clouds, vanishing points) can only be used for recovering orientation. However, since orientation is already covered by the particle filter, and is integrated over time, the benefit of using far-away points for recovering orientation is limited.

4.5 Experiments

Images from the major urban train station of Philadelphia were used for these experiments. The dataset consists of 20 thousand omnidirectional images (360-degree field of view) captured using a PointGrey Ladybug 3 camera. Like the Ladybug used in chapter 3 the unit consists of six cameras mounted in a hemi-sphere with five cameras in a circle and the sixth camera pointing upwards. Again, only images from the five lateral cameras were use. The full image set consists of 100 thousand images. The camera was mounted on a differential-drive vehicle and driven around the environment.

The train station is a large indoor environment containing both large open spaces and small spaces, such as hallways, shops, restaurants, and booths. Being primarily a pedestrian environment, motion was unconstrained. Several portions of the station were traversed multiple times, approaching previously visited locations from different and opposite directions. This is in contrast to data captured with an outdoor vehicular setup which is often restricted to retracing its path, visiting previous locations with identical trajectories.

In order to emphasize localization in large open indoor spaces, results for the station's main hall are presented. This room is 88 by 41 meters and 29m high. About 40 percent of the image data, around 8,000 views, were captured within this area. These are sampled at a rate of 1-in-10 resulting in a final video sequence of 791 views (resulting in an image approximately every 2m).

4.5.1 Object Detection

For the purposes of localization, the map of the environment needs to be populated with the locations of the objects. It is assumed that the map being used is up-to-date, as such, all objects were selected objects only if they are permanent fixtures (e.g. clocks, payphones) or are unlikely to move significant distances (e.g. trashcans). The final set of objects employed is: trashcan, clock, payphone, ticket machine. Each object template was constructed by a single object exemplar. Both histograms of oriented energies, and histograms of quantized colors are computed the resulting product used to perform detection.

Object detection in this setting is especially challenging due to the method the images were captured. The fact that the camera was moving precluded high-quality capture of images from the lateral cameras due to motion blur (figure 4.1). Additionally, this motion also prevented objects from being clearly centered in the camera images (figure 4.1), a condition that is commonly assumed in traditional object recognition to facilitate detection.

A *single example* was selected for each object category. Employing a single positive example eliminates the need for extensive labeling and training that is common with most approaches [Bishop, 2006; Felzenszwalb et al., 2010]. This results in a simpler detector with almost no offline pre-processing. The primary disadvantage is complete lack of object generalization, the ability to handle intra-class variation [3]. Although the ability of an object detector to handle intra-class variation is critical for the general task of object detection, it is not a necessity for the localization problem.

Of the categories chosen, the clock has the most peaky heatmaps. Its distinctive color, white, and very clear boundary combine to yield isolated hot spots (figure 4.4 top row). The ticket machines (figure 4.4 second row) are generally installed in groups; this confused the HQC feature creating "warmer" regions around the objects. Note however that clearly

[3] Intra-class variation is the variation in appearance between two object instances belonging to the same object category, e.g. two ATMs of different banks.

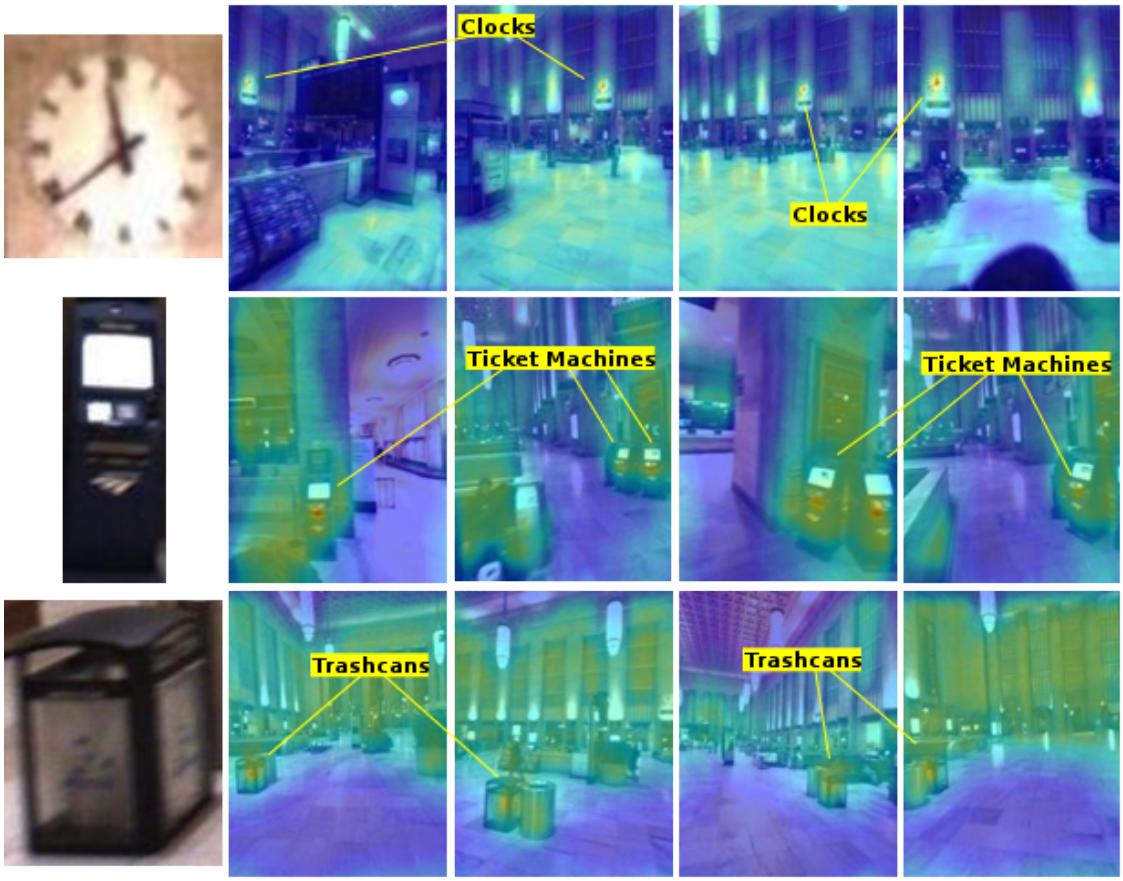


Figure 4.4: Detection results, with the object template in the first column and resulting heatmaps in the remaining columns. Hot spots, orange to red, indicate increased object presence (Best viewed in color).

distinct red "hot" spots appear centered on the machines. The detections for the trashcan are somewhat weaker (figure 4.4 bottom row) with lukewarm peaks in the heatmaps. These are caused by occlusions, transparencies, and perspective distortion. In some cases, the trashcan is partially obscured, lowering the match scores. Furthermore, the transparencies caused by the plastic bags covering most of the trashcan template were not taken into account As such, both objects behind the trashcan (figure 4.2c) and its content adversely affect the matching, especially with respect to color histograms. Furthermore, a distinct disadvantage of this single-exemplar object model is its inability to recognize the trashcan when viewed from a different perspective. This shortcoming could be addressed by

adding multiple templates for multiple views of each object. This would indeed increase performance but also the time required to detect objects, and require finding and extracting multiple exemplars for each object to cover multiple angles. However, a full coverage of each object is not necessary for localization as robot location information is integrated over time. The histogram based approach is still able to detect the presence of the trashcan even when flipped with a single exemplar. By taking advantage of partial symmetry in the objects, we can compute heatmaps using flipped templates and select the maximal response from the original and mirrored templates.

Other considerations that lead to reduced detection performance include extreme lighting, low resolution imagery, and scene clutter. Although the desired object is clearly visible (figure 4.5a), a combination of low resolution and lighting change yield a very low match score resulting in a cold spot, an example of perceptual variability. On the other hand, for perceptual aliasing, the alignment of a person with dark clothing with a white advertisement creates the hallucination of a ticket machine (figure 4.5b).

4.5.2 Localization

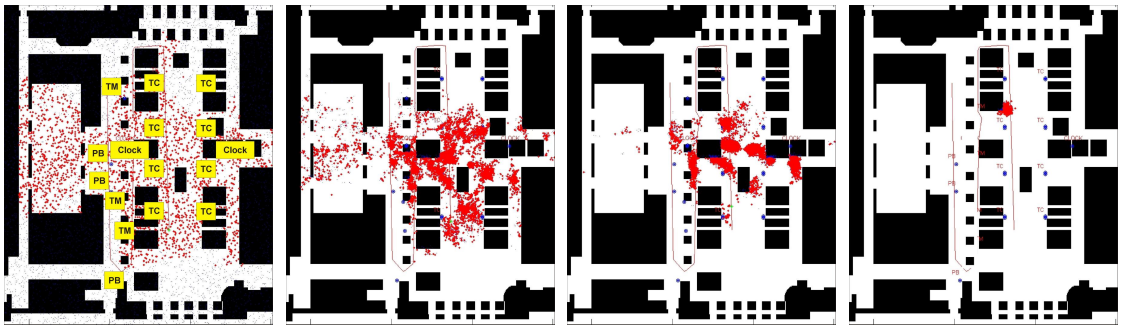
Getting accurate ground-truth data in an indoor setting is a challenging problem in itself. Key frames in the video sequence were annotated with their ground truth position information during collection on a simplified building schematic. These were then used to interpolate position information for all remaining images.

In order to use objects for localization, a map of the train station was manually annotated with objects and their locations. Objects were treated as points, covering no area. This did not greatly affect localization results, as the objects in question all have small footprints.

The map of the station with the object is showcased in figure 4.6a. The red line indicates the hand-labeled ground-truth camera trajectory, with the green star denoting the current ground-truth position. 10,000 particles are used, denoted with red dots. Note how the uniform particle cloud coalesces into clusters (figure 4.6b). Multiple instances of each

(a) (b)

Figure 4.5: Some detection failures: (a) Missed object, resulting in a cold spot. (b) Hot area resulting from object hallucinations.

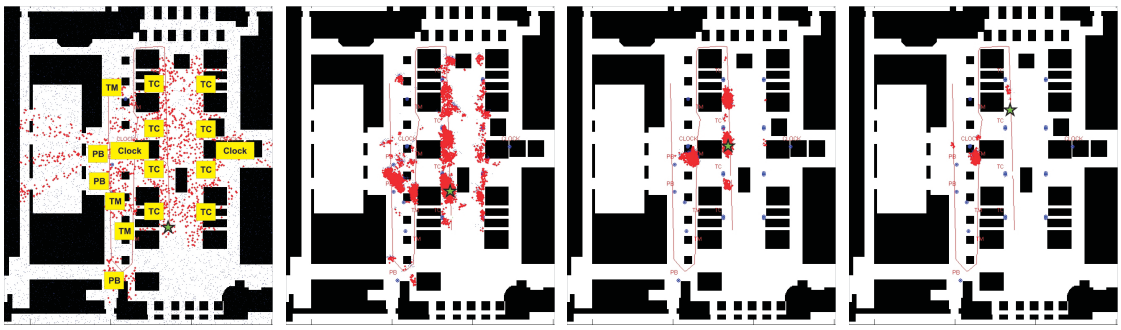


(a) (b) (c) (d)

Figure 4.6: Global localization results using the proposed soft object detector. Particle locations at (a) one, (b) ten, (c) twenty, and (d) forty iterations. (a) The objects used for localization are labeled in the positions where they appear in the map: clock, trashcan (TC), phone booth (PB), ticket machine (TM).

object category creates ambiguity generating multiple location hypotheses (figure 4.6c). Eventually, enough objects are observed that the system focuses around the true location (figure 4.6d). To demonstrate the advantages of soft object detectors localization is also performed using a "hardened" version of the detector. In order to generate traditional bounding boxes from the soft detector, the projected heatmaps are thresholded (figure 4.8 blue) with a fixed threshold (chosen appropriately for each object category). Then non-maxima suppression is applied to find local maxima in the response map; these determine the positive detections (figure 4.8 green). Each local maxima represents a positively detected bounding box, with a fixed width. The final signal fed into the localizer is a binary one-dimensional heatmap with value 1 in the direction where there exists a positive detection, and 0 elsewhere (figure 4.8 red).

The use of a more traditional hard detector has a strong clustering effect on the particles. Starting even at one iteration (figure 4.7a) the particles are noticeably less spread out. Although this provides increased confidence in the computed position with smaller, tighter clusters, it is more susceptible to incorrect detections, in the end failing to correctly localize the camera (figure 4.7d).



(a) (b) (c) (d)

Figure 4.7: Global localization results using a hard object detector. Particle locations at (a) one, (b) ten, (c) twenty, and (d) forty iterations.

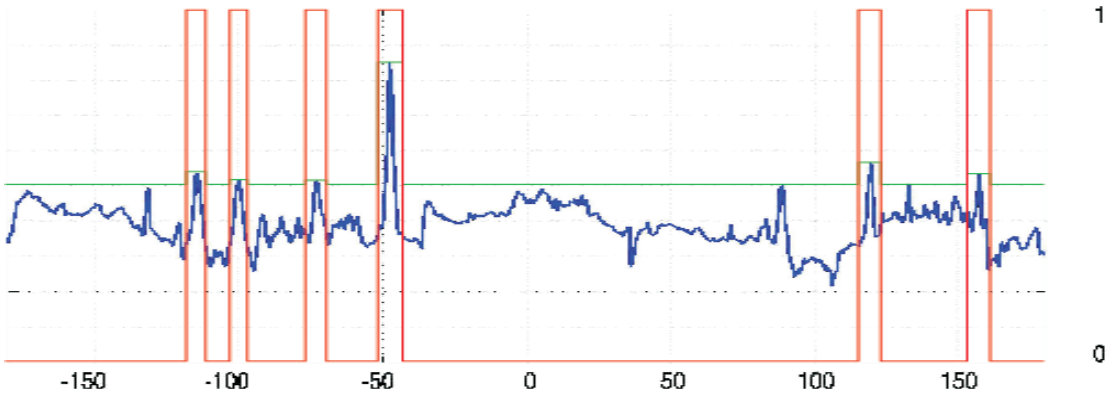


Figure 4.8: Creating a Hard Detector: Original heatmap (blue), with fixed thresholding and non-maxima suppression (green) resulting in a binary signal (red).

The ability of soft detections to include weak signals prevents from over-committing to an incorrect localization. This comes at the cost of larger uncertainty in the positioning leading to larger particle clusters. One could argue that a similar effect could be achieved by inserting random particles at each iteration of the localization process, adding random noise. This system takes a more systematic approach, incorporating the uncertainty at the source of the detection (where it originates) rather that in the processing.

4.5.3 Discussion

Localization was performed in a large open environment. Instead of using a traditional object detector, soft-object detection is used to generate heatmaps. The dense heatmaps are never thresholded, assuring that weak signals are not dismissed during localization. Instead, localization is driven by the similarity between the observed and expected heatmaps. The latter are created by mixing Gaussians at bearings where objects are expected (equation (4.7)).

This formulation does not consider missing objects when computing object layout similarity between locations (equation (4.6)). Although disregarding negative data in the form of missing objects provides some robustness, it weakens the system with regards to perceptual aliasing. By not taking into account all objects in the scene (either visible but unexpected, or expected and not visible), potential cues differentiating between locations are missed. On the other hand, the inclusion of negative data would significantly increase susceptibility to perceptual variability.

Accounting for and penalizing for negative data will lower the similarity score due noise in the observed signal. Accurately simulating this noise would require having dense visual representation of the environment. This goes counter to the purpose of this chapter: localization based on semantic annotations. On the other hand, the introduction of generic noise to in the simulated signal to compensate for this would constitute a threshold on the heatmap.

In order to address this short-coming, negative data should be explicitly modelled.

This require a more robust object detector, better able to deal with perceptual aliasing. Section 4.6 shows how a traditional object detector is modified to generate heatmaps. This enables the use of state-of-the-art object detectors for the purposes of soft-detection based localization.

Incorporating a more robust detector allows explicit modelling of objects in the scene, incorporating both negative and positive cues. Section 4.7 re-examines the particle filter formulation from section 4.4 and expands on it. Significantly the perception update term section 4.4.2 is re-formulated using a generative model. Unlike equation (4.6), using a generative model allows use of both negative and positive signals from the data.

4.6 Heatmaps from Object Detectors

The detection framework described in section 4.3 although sufficient for the purposes of localization is not robust. Based on a single template image and combining both color and gradient it is able to detect and localize when there is no intra-class variability. By matching templates across an entire image it naturally resulted in heatmaps and was well suited as a soft object detector.

In order to extend localization to more challenging scenes a more robust detector is needed. This will allow localization when object appearance is inconsistent, either due to object pose or intra-class variability. However traditional detectors, which usually focus in individual bounding boxes, don't provide the requisite heatmaps. The task then is two-fold: find robust object detector and adapt it to generate heatmaps.

4.6.1 Deformable Parts Model

Discriminatively trained deformable part models (DPM) [Felzenszwalb et al., 2010] is a robust multi-scale object detector. The fundamental building block of DPM is local image features computed over image patches. Filter responses are computed for these features at multiple locations and scales. The score of a sub-window in an image is then the sum of

these responses.

To model object deformations, a *component* is scored using filter responses computed at two levels. First, a coarse root filter is used to localize the object as a whole. Then fine-grained part filters are used to score object parts at higher resolutions. Using higher resolution filters results in more accurate and reliable localization of parts. Furthermore, part filter locations are flexible within the root window, explicitly allowing object deformations. The final score combines both appearance (filter responses) and geometry (relative part locations). The model is made more robust by the inclusion of multiple independent *components* per object class.

DPM models are trained as latent SVMs, with part locations and filter weights as latent variables. Using a labeled set of positive examples, optimization proceeds in two steps. First, latent variables are maximized for positive examples; then, model parameters are optimized over both positive and negative examples. Instead of using all the negative examples, hard-negatives are mined from the training data by looking for the highest scoring false-positives. The process is repeated for a number of iterations.

The resulting 1-vs-all classifier is then used to detect objects in an image. A feature pyramid is generated for a query image, and each component of a model is evaluated in a sliding window fashion. Filter responses are densely computed both for root filters and part filters. Responses to part filters are also transformed to allow for spatial uncertainty (i.e. deformation). The final score at any location and scale is the sum of it's root score and the maximal parts' scores. Scores are computed for all components in a model and thresholded. Finally greedy non-maxima suppression is used to remove significantly overlapping detections.

4.6.2 Extracting Heatmaps

Like most object detectors, DPM is designed to localize and identify individual object instances in an image. As mentioned earlier, the DPM output is short list of bounding boxes with scores and component ids. However, for our purposes we're interested in

object-heatmaps with dense per-pixel scores. In order to generate these heatmaps from DPM detections, the following steps are taken:

1. DPM Detections: DPM detections are computed as usual except two changes. Since the goal is to collect scores for all image pixels, both filtering mechanisms from DPM are turned off. Instead of using the DPM model threshold (or some other trained threshold) a threshold of $-\infty$ is used. Further, once all bounding boxes are extracted the non-maxima-suppression step is skipped. Together these yield a dense sampling of DPM scores across the entire image at multiple scales.

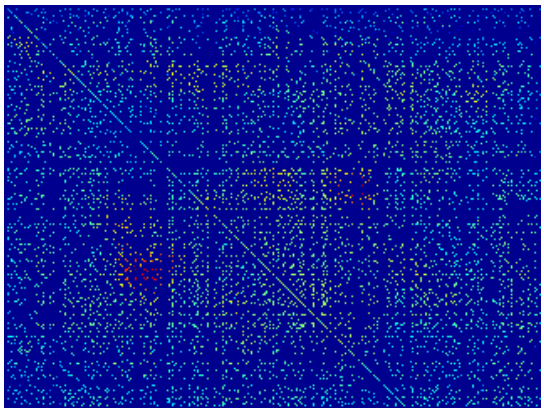
2. Centering and Aggregation: For each detected bounding box, its center pixel is computed. Note that multiple detections can have equivalent centers due to either multiple-scales or different root components. Each pixel in the image is then assigned the maximum score among all detections centered on it.

3. Interpolation: Even with filtering turned off, DPM still doesn't compute a score for every pixel in the image. Therefore scores are assigned for all pixels missing them using nearest neighbor interpolation.

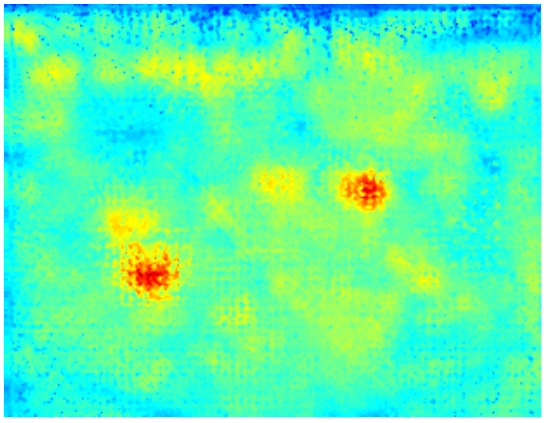
This results in a dense heatmap which generates soft object-detections from a classic object detector. An example is shown in figure 4.9.

Here a DPM model is trained to detect soda-cans. Using the standard DPM procedure results in a single detection and bounding box (figure 4.9a) resulting in a false negative. The heatmap (figure 4.9c) on the other hand has two hot-spots. Figure 4.9d overlays this heatmap on top of the original image. Once heatmaps are computed for an image, they can be fed into the particle filter for localization.

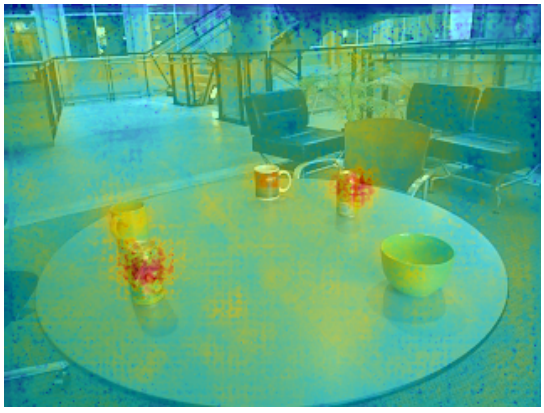
(a) DPM Bounding Boxes



(b) Unfiltered DPM Detections



(c) Interpolated DPM Heatmap



(d) Heatmap Overlay

Figure 4.9: Generating heatmaps from DPM. (a) DPM bounding boxes for object class *soda_can*, (b) unfiltered DPM detections, and (c) the final heatmap. (d) shows heatmap (c) overlayed on top of the original image.

4.7 Perception Update (Revisited)

Recall that the particle filter in section 4.4 estimates $p(\mathbf{x}_t \mid Z_{1:t}, M)$, the probability of position $\mathbf{x}_t$ at time t given map M and all the observations from time 1 to time t: $Z_{1:t} = \{z_1, \ldots, z_t\}$.

In that section, the particle estimate was composed of two parts: the motion update equation (4.5) and the perception update equation (4.6). Specifically, the weight of each particle is the sum of the inner products between the observed and the expected heatmap of each object category,

$$w^i = \sum_{j=1}^{n} < \mathbf{z}_j^i, \hat{\mathbf{z}}_j^i >. \tag{4.6}$$

Although simple to implement, this formulation only makes use of positive correlation between the heatmaps. Furthermore it's an indirect cue, measuring the similarity between the observation and a simulation without modelling the underlying process.

As a pathological example, consider an observed heatmap that is all zero, meaning the object detector found no objects. Consider two particles, the first is located near objects, and has an expected heatmap with several peaks, the second is far from all objects and has an all zero heatmap. Using the original formulation, the likelihood of both particles is equal and zero, measuring the correlation between non-existent peaks. However, it's easy to see that the likelihood of the second particle should be higher. The lack of observations imply there are no objects nearby, yet this isn't incorporated into the particle likelihood.

To address these shortcomings, this section introduces a novel formulation of the perception update that explicitly models objects in the scene. It uses a generative score model for the object detector incorporating both negative and positive cues.

4.7.1 Probability of Position

The use of motion update and perception update in the particle filter stems directly from the target distribution $p(\mathbf{x}_t \mid Z_{1:t}, M)$. This probability of can be split into three components,

$$\begin{aligned} p(\mathbf{x}_t \mid Z_{1:t}, M) &= p(\mathbf{x}_t \mid z_t, Z_{1:t-1}, M) \\ &= \frac{1}{p(z_t, Z_{1:t-1}, M)} p(z_t \mid Z_{1:t-1}, \mathbf{x}_t, M)\, p(Z_{1:t-1}, \mathbf{x}_t, M) \\ &= \frac{1}{p(Z_{1:t}, M)} p(z_t \mid Z_{1:t-1}, \mathbf{x}_t, M)\, p(Z_{1:t-1}, \mathbf{x}_t, M)\,. \end{aligned} \tag{4.9}$$

The Markov assumption is used here in two ways:

Assumption 1. Observation data z_t given the current position $\mathbf{x}_t$ and map M is conditionally independent of everything else.

$$p(z_t \mid Z_{1:t-1}, \mathbf{x}_t, M) = p(z_t \mid \mathbf{x}_t, M)\,. \tag{4.10}$$

Assumption 2. Given the past position and the map $\mathbf{x}_{t-1}$, the current position $\mathbf{x}_t$ is conditionally independent of past observations.

$$p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, M, Z_{1:t-1}) = p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, M) \tag{4.11}$$

In order to use the second assumption, the third term is marginalized over past positions $\mathbf{x}_{t-1}$,

$$\begin{aligned} p(Z_{1:t-1}, \mathbf{x}_t, M) &= \sum_{\mathbf{x}_{t-1}} p(Z_{1:t-1}, \mathbf{x}_t, \mathbf{x}_{t-1}, M) \\ &= \sum_{\mathbf{x}_{t-1}} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, M, Z_{1:t-1})\, p(\mathbf{x}_{t-1}, Z_{1:t-1}, M) \\ &= \sum_{\mathbf{x}_{t-1}} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, M, Z_{1:t-1})\, p(\mathbf{x}_{t-1} \mid Z_{1:t-1}, M)\, p(Z_{1:t-1}, M) \\ &= p(Z_{1:t-1}, M) \sum_{\mathbf{x}_{t-1}} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, M, Z_{1:t-1})\, p(\mathbf{x}_{t-1} \mid Z_{1:t-1}, M)\,. \end{aligned} \tag{4.12}$$

Combining equations (4.9) to (4.12) together results in,

$$p\left(\mathbf{x}_t \mid Z_{1:t}, M\right) = \frac{p\left(Z_{1:t-1}, M\right)}{p\left(Z_{1:t}, M\right)} p\left(z_t \mid \mathbf{x}_t, M\right) \sum_{\mathbf{x}_{t-1}} p\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}, M\right) p\left(\mathbf{x}_{t-1} \mid Z_{1:t-1}, M\right). \tag{4.13}$$

The first term, $\frac{p(Z_{1:t-1},M)}{p(Z_{1:t},M)}$ is a normalizing factor independent of $\mathbf{x}_t$. $p\left(z_t \mid \mathbf{x}_t, M\right)$ is the observation likelihood term, and forms the basis of the perception update. The motion model, $p\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}, M\right)$, is the basis of the motion update. Finally, $p\left(\mathbf{x}_{t-1} \mid Z_{1:t-1}, M\right)$, is the prior distribution, modeled by the iterative nature of the particle filter.

4.7.2 Observation Likelihood

In order to solve

$$p\left(z_t \mid \mathbf{x}_t, M\right), \tag{4.14}$$

and explicitly include objects as part of the observation likelihood, the perception process needs to be modelled.

As before, the observation z_t (The time variable t will be omitted going forward) is a collection of object detector response scores. For a set of k object classes,

$$z = \{z^1, \ldots, z^k\}, \tag{4.15}$$

with z^i the detector scores for a single object class. Each z^i contains a heatline, the scores for each column in the image for detector i. Rather than frame score values in image coordinates, each column is mapped to a relative bearing θ.

Similarly, it would be useful to consider $\mathbf{x}$ and M in the same frame as z^i using relative bearings. The map is defined by a collection of objects $M = \{m_1, \ldots m_n\}$ with each $m = (x, y, c)$ containing position and class information. It's possible to define a coordinate transform $T : \mathbf{x}, m \mapsto \mathbf{y}$ which converts an absolute object instance m to a relative instance $\mathbf{y}$ with respect to position $\mathbf{x}$. $\mathbf{y} = (\theta, c)$, includes the relative angle θ between object m and position $\mathbf{x}$ and the object class c. This re-frames the observation model equation (4.14),

$$p\left(z \mid \mathbf{x}, M\right) = p\left(z \mid Y\right), \tag{4.16}$$

where $Y = \{\mathbf{y}_i = T(\mathbf{x}, m_i) \mid i = [1 \dots n]\}$.

In order to factor this term out further, two additional assumptions are made:

Assumption 3. Observation likelihood is independent across object classes.

This requires that the underlying object detector be sufficiently discriminative across object classes. Detector scores for detector i on objects of class j should not yield *unusually* high scores. This is achieved during training by including examples of objects of class j as negative examples when training detector i. High scores for objects of class j in detector i are then be part of the model, and no longer *unusual*. This results in the following equation:

$$p\left(z \mid Y\right) = \prod_{c} p\left(z_c \mid Y^c\right), \tag{4.17}$$

where $Y^c = \{\mathbf{y}_i = (\theta_i, c_i) \mid c_i = c\}$, is the subset of Y of objects of class c.

Assumption 4. Given Y, scores for different columns are independent.

Since Y contains the objects being detected, it directly informs the scores for every column, making scores independent across columns. This yields:

$$p\left(z \mid Y\right) = \prod_{\theta} p\left(z(\theta) \mid Y\right). \tag{4.18}$$

Applying equations (4.14) to (4.18) to equation (4.14) results in

$$p\left(z_t \mid \mathbf{x}_t, M\right) = \prod_{c} \prod_{\theta} p\left(z_t^c(\theta) \mid Y_t^c\right), \tag{4.19}$$

which shows that the scores for a set of observations z_t can be factored into individual scores for each class and column $z_t^c(\theta)$. The method of computing individual column scores is discussed next.

4.7.3 Generative Score Likelihood

In order to evaluate the probability of an individual score value, $p\left(z_t^c(\theta) \mid Y_t^c\right)$, a generative model is used. Since this section deals with individual score values for a single class, t and

c are temporarily dropped from the notation for simplicity. If this model is built from a separate set of training data, it needs to be decoupled from the map and position currently under evaluation. To separate the score from the localization context, another variable is introduced. Specifically, a score z could derive from a positive or negative detection,

$$p\left(z(\theta) \mid Y\right) = p\left(z(\theta), d(\theta) \mid Y\right) + p\left(z(\theta), \bar{d}(\theta) \mid Y\right), \tag{4.20}$$

where $d(\theta)$ indicates an object is present at bearing θ, while $\bar{d}(\theta)$ indicates no object present at θ.

Factoring out the terms,

$$p\left(z(\theta), d(\theta) \mid Y\right) = p\left(z(\theta) \mid d(\theta), Y\right) p\left(d(\theta) \mid Y\right), \tag{4.21}$$

will allow separating the score z from the geometry Y Since now the actual event of a detection or non-detection is accounted for explicitly, another independence assumption can be used.

Assumption 5. Given the actual detection determination $d(\theta)$ or $\bar{d}(\theta)$, the score $z(\theta)$ is independent of the objects Y,

Once the presence or absence of an object is made, the layout of the scene no longer has any bearing on the actual score,

$$p\left(z(\theta) \mid d(\theta), Y\right) = p\left(z(\theta) \mid d(\theta)\right) = p\left(z \mid d\right). \tag{4.22}$$

Combining this with equation (4.20) results in

$$p\left(z(\theta) \mid Y\right) = p\left(z \mid d\right) p\left(d(\theta) \mid Y\right) + p\left(z \mid \bar{d}\right) p\left(\bar{d}(\theta) \mid Y\right), \tag{4.23}$$

where

$$p\left(z \mid d\right) \text{ and } p\left(z \mid \bar{d}\right) \tag{4.24}$$

are the score likelihoods learned from training data and,

$$p\left(d(\theta) \mid Y\right) \text{ and } p\left(\bar{d}(\theta) \mid Y\right) \tag{4.25}$$

are detection likelihoods from scene geometry.

The detection likelihoods are modelled using a normal distributed similar to Atanasov et al. [2014, Eqn. 2]. There, explicit object *instance* associations are made, whereas this work marginalizes over all instances within an *class*. Define

$$\begin{aligned} p\left(d(\theta) \,|\, Y\right) &= \max_{\mathbf{y}_i \in Y} p\left(d(\theta) \,|\, \mathbf{y}_i\right), \text{and} \\ p\left(\bar{d}(\theta) \,|\, Y\right) &= 1 - p\left(d(\theta) \,|\, Y\right), \end{aligned} \tag{4.26}$$

where $p\left(d(\theta) \,|\, \mathbf{y}_i\right) \sim \mathcal{N}(\theta_i, \sigma)$ where $\mathbf{y}_i = (\theta_i, c)$ and σ is a field of view threshold (in this case $5°$).

As mentioned earlier, the score likelihoods are learned from the training data. Given a collection of labeled detections kernel density estimation (KDE) is used to generate score distributions. Positive detections for $p\left(z \,|\, d\right)$ and negative detections for $p\left(z \,|\, \bar{d}\right)$. The distribution is sampled at 1000 points using a normal kernel. The kernel bandwidth is taken as the robust estimate of the sample standard deviation assuming a normal distribution for the scores.

4.7.4 Particle Weights

Now that the observation likelihood has been fully specified,

$$p\left(z_t \,|\, \mathbf{x}_t, M\right) = \prod_c \prod_\theta p\left(z_t^c(\theta) \,|\, d(\theta)\right) p\left(d(\theta) \,|\, Y_t^c\right) + p\left(z_t^c(\theta) \,|\, \bar{d}(\theta)\right) p\left(\bar{d}(\theta) \,|\, Y_t^c\right), \tag{4.27}$$

it can be used for the perception update step of the particle filter in lieu of equation (4.6):

$$w^i = p\left(z_t \,|\, \mathbf{x}_t = s_t^i, M\right), \tag{4.28}$$

where w^i is the weight of the particle s_t^i at time t.

The next section includes localization experiments with the particle weights computed using equation (4.28).

4.8 Additional Experiments

In order to showcase localization with the extended perception update formulation the RGBD Scenes Dataset v2 Lai et al. [2014] is used. This dataset contains a similar scenario to section 4.5. There is a moving camera in an open scene with multiple instances of multiple object classes. Furthermore the dataset includes a robust collection of object training images required to train the DPM object detectors.

The RGBD Scenes Dataset v2 Lai et al. [2014] consists of 14 video sequences captured with a Microsoft Kinect device. In each scene, there is a surface (table or counter) with several objects (e.g. coffee mug, soda can). The camera moves around the scene, viewing the object from various angles. For these experiments, five classes of objects are used: bowl, cap, cereal box, coffee mug and soda can. Details for each scene are in table 4.1.

In addition to the RGB images, the dataset also includes a labeled point clouds and camera trajectories. The point clouds are generated from the Kinect ToF (time-of-flight) sensor, with each point labeled with the class of the object is falls on. The point cloud was used solely for generating the object map used for the localization experiments. Localization results are evaluated with respect to the provided camera trajectory.

The RGBD Objects Dataset [Lai et al., 2011b] was used to train a discriminative object detector. These include multiple instances from each object class on a turntable, with images captured from various viewpoints. This dataset also includes manual bounding boxes for each object in every image. A summary of this dataset is provided in table 4.2.

4.8.1 Object Detection

Individual DPM models were trained for every object class using the code provided by Felzenszwalb et al.. Training data included all the images from Lai et al. [2011b] for the relevant classes. The model was trained in a 1-vs-all manner, with images for all other classes used as negative examples for the class under training. Unfortunately these models were unable to cope with the changes in lighting and perspective between the images in

Scene ID	No. Images	Object Counts				
		Bowl	Cap	Cereal Box	Coffee Mug	Soda Can
1	888	2	1	0	1	1
2	834	1	0	1	1	2
3	861	2	1	0	1	1
4	868	2	0	1	1	1
5	1128	1	1	1	1	0
6	1048	1	0	0	2	2
7	943	1	1	1	1	1
8	925	0	2	0	0	1
9	732	1	0	1	0	1
10	716	1	1	0	1	0
11	640	0	0	1	0	2
12	723	0	2	0	1	0
13	462	1	0	1	1	1
14	659	1	1	0	1	1

Table 4.1: Summary of RGBD Scenes v2 Dataset. Those scenes used to train the object detector are highlighted.

Object Class	No. Examples	No. Images
Bowl	6	3884
Cap	4	2551
Cereal Box	5	2929
Coffee Mug	8	4866
Soda Can	6	3555

Table 4.2: Summary of RGBD Objects Dataset

Lai et al. [2011b] and Lai et al. [2014]. To compensate for this, four scenes from Lai et al. [2014] (No. 6,7,11 and 12) were used to train a new set of DPM models. Bounding boxes for the objects were extracted from the labelled point cloud data included in Lai et al. [2014]. Those scenes were chosen so that each object class is present in at least two scenes, and *missing* from at least one scene (for negative examples). These are highlighted in table 4.1. The resulting DPM models performed significantly better.

Figure 4.10 shows precision-recall curves for both models for all scenes except those used in training the DPM (No. 6,7,11 and 12). The ground-truth bounding boxes used to compute precision and recall were generated using labeled point-cloud data included in Lai et al. [2014]. The labeled point cloud was projected into each image using the ground-truth camera position. For each object class individual pixels were clustered using k-means clustering with k set to the number of class instances in the scene. A bounding box is drawn around each cluster and used as reference. Boxes with either width or height less than 10 pixels were discarded.

Every DPM model was evaluated on each image using a threshold of $-\infty$. Non-maxima suppression is used to eliminate duplicate detections with a threshold of 0.5. Detections are considered true-positives if they cover at least 50% of the area of a ground-truth box, and are false positives otherwise. A false negative occurs when no detection meets this criteria for each given ground-truth box.

The initial models trained solely on Lai et al. [2011b] and labeled *objects* are on the left. The models trained using images from both Lai et al. [2011b] and Lai et al. [2014] are labeled *object + scenes* and are on the right.

4.8.2 Score Likelihoods

The generative score likelihood models were generated using the same scenes used to train the detector (No. 6,7,11 and 12). Like before, every DPM model was evaluated on each image using a threshold of $-\infty$. The whole set of detections was used, and non-maxima suppression was skipped. Detection were labeled as before, with positive

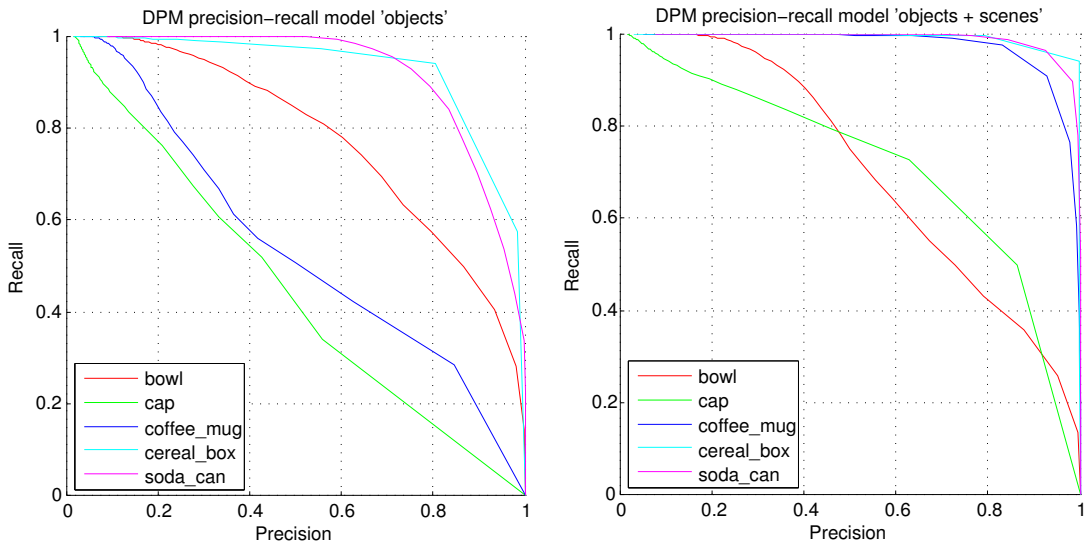


Figure 4.10: Precision-recall curves comparing DPM models trained using only images from Lai et al. [2011b] (left) or using images from both Lai et al. [2011b] and Lai et al. [2014] (right).

detections covering at least 50% of the area of a ground-truth box.

Score likelihoods are assumed to be normally distributed. Kernel density estimation was used to sample the distribution, with 1000 samples and a normal kernel. The kernel bandwidth was set from the robust standard-deviation of each sample set.

Figure 4.11 contains the estimated PDF for each likelihood distribution for both models. Unsurprisingly the distributions for the *objects* model have much greater overlap than the *objects* + *scenes* distributions. Specifically the score-for-detection distributions in the *objects* model (dotted green line) are much further to the left. The greater amount of overlap implies the *objects* trained detector is less discriminative. Note that score-for-non-detection rarely yield positive scores, in part a result of the DPM hard-negative mining procedure.

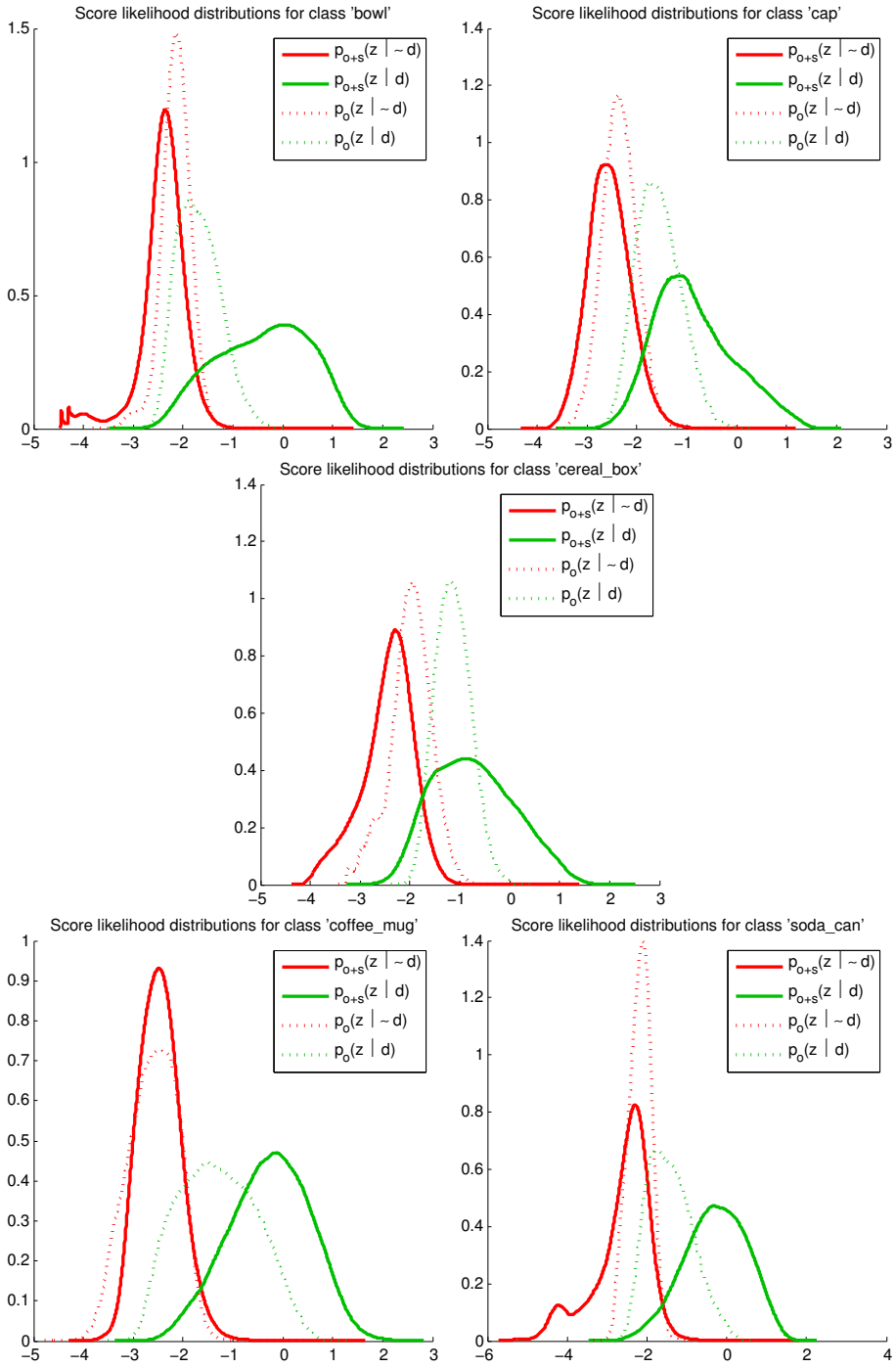


Figure 4.11: PDF of the score likelihood functions for each object class. Score distributions for negative detections in red, and for positive detections in green. The dotted line is the *objects* model, while the solid line is the *objects + scenes* model.

4.8.3 Simulations

In addition to localization results using the trained object detectors, two sets of simulations are also included. For both of these variations the heatlines z^t are generated from ground-truth data. The first simulation, *synthetic-scores*, uses synthetic score distributions in the range $[0, 1]$:

$$p\left(z \mid d\right) \sim \mathcal{N}(0.75, 0.25), \text{ and } p\left(z \mid \bar{d}\right) \sim \mathcal{N}(0.25, 0.25).$$

Additionally, heatlines are generated from ground-truth object bounding boxes B. For each bounding boxes $b_i = [x_i, y_i, w_i, h_i, c_i]$ a heatline $\hat{z}_i$ is generated from an un-normalized Gaussian centered at x_i with std-dev w_i. A small noise floor of 0.01 is added to avoid completely degenerate heatlines.

$$\hat{z}_i(\theta) = e^{-\left(\frac{\theta - x_i}{w_i}\right)^2}.$$

The max is taken across all instances for the same class,

$$\hat{z}^c = \max\left(\max_{b_i \in B | c_i = c} \hat{z}_i, 0.01\right).$$

The resulting heatlines and localization results are completely independent of both object detection process and the DPM models.

The second simulation, *simulated-DPM*, uses the score distributions from section 4.8.2 but skips the DPM detection step. Here,

$$\hat{z}_i(\theta) \sim \begin{cases} p\left(z \mid d\right) \text{ if } |\theta - x_i| \leq w_i \\ p\left(z \mid \bar{d}\right) \text{ if } |\theta - x_i| > w_i \end{cases}. \tag{4.29}$$

This results in heatlines that are based on the learned DPM models, but are independent of the object detection process.

4.8.4 Localization

Localization results are presented for the first 12 scenes in Lai et al. [2014]. The underlying object maps for scenes 13 and 14 were not recovered correctly. Without relevant maps localization cannot be performed.

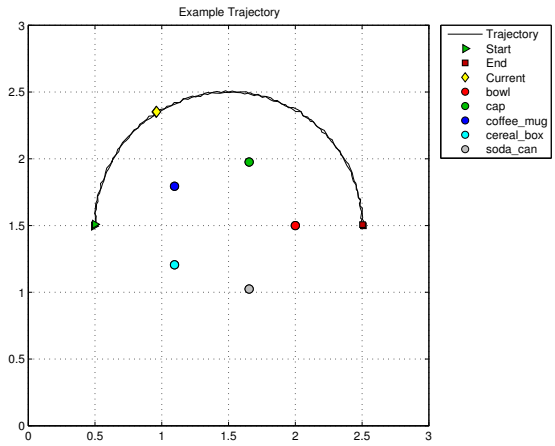
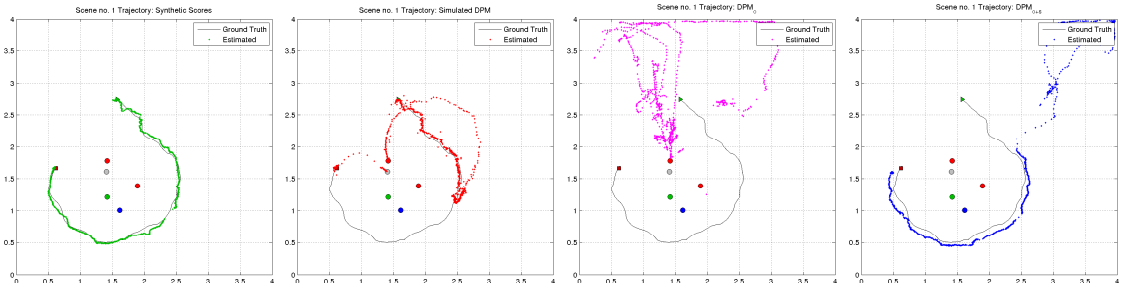


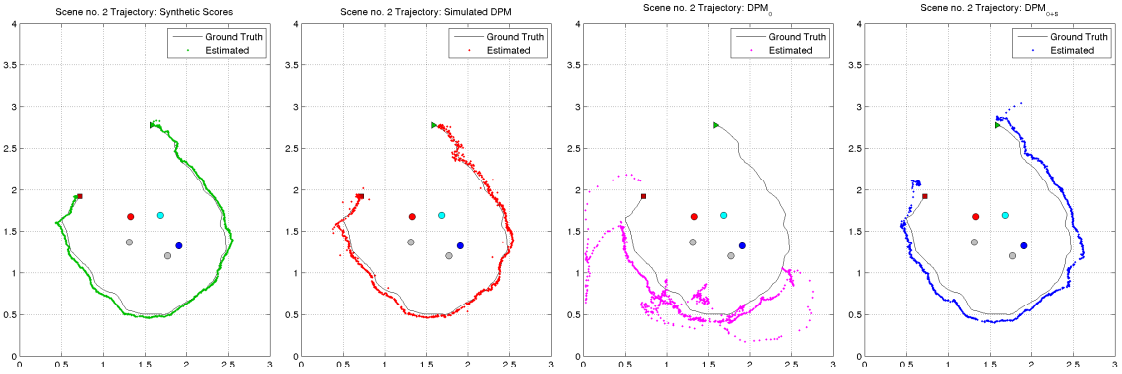
Figure 4.12: RGBD Scenes Legend. Each scene from [Lai et al., 2014] contains a trajectory (black) starting at the green marker and ending at the red marker. Objects are marked by solid circles, colored by their object class. These colors are maintained throughout this chapter.

Results are included for both DPM models: *object* (DPM_o) and *objects + scenes* (DPM_{o+s}) and simulations *synthetic-scores* and *simulated-DPM* defined in section 4.8.3. Figure 4.12 shows a toy scenario with a trajectory and objects. This layout and colors are used in figure 4.13 and other figures below, and serves as a legend.

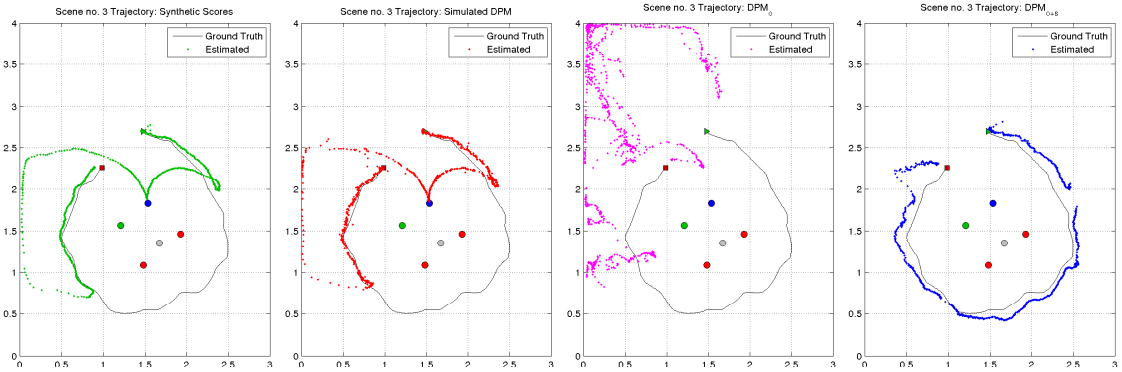
Figure 4.13 includes recovered trajectories for all models and scenes. Time-series of the root-mean-square of the localization error are presented in section 4.8.4. There are several categories of localization failures: detection errors, not enough objects, occlusions and geometric ambiguity. These affect the different models in various ways.



(a) Estimated trajectories for scene 1

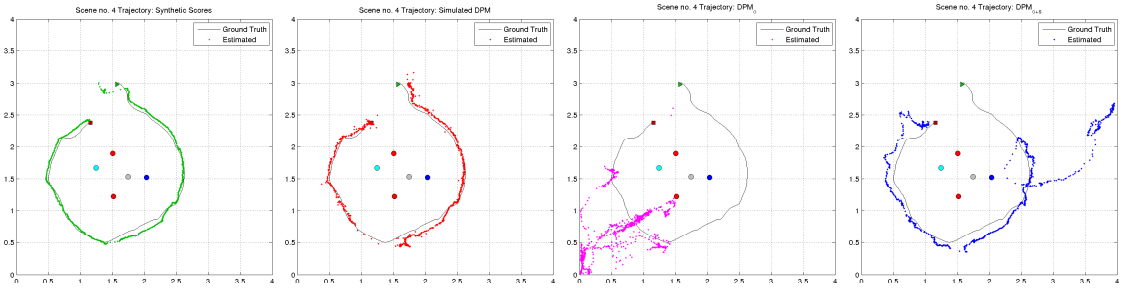


(b) Estimated trajectories for scene 2

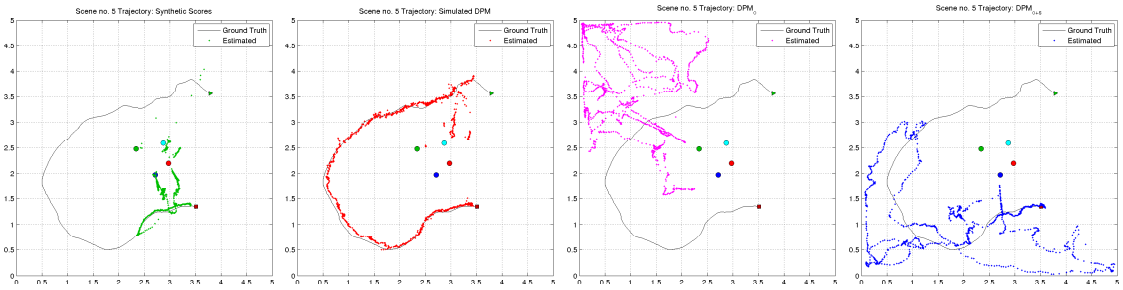


(c) Estimated trajectories for scene 3

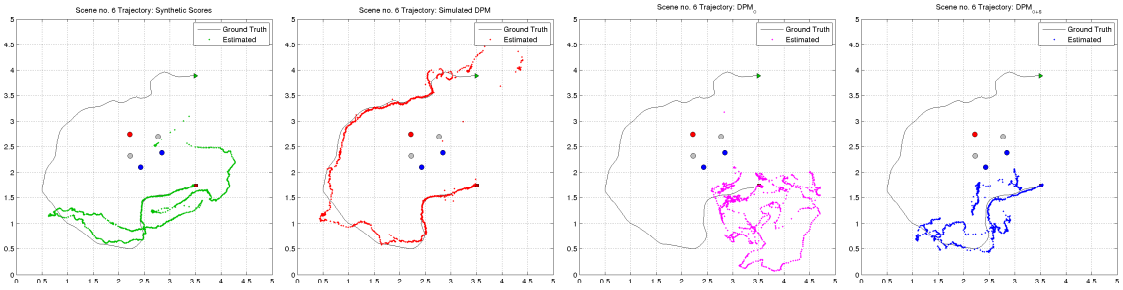
Figure 4.13: Estimated trajectories. Position at each iteration computed from weighted average position of particles. From left to right: *synthetic-scores*, *simulated-DPM* , *objects*, *objects + scenes*.



(d) Estimated trajectories for scene 4

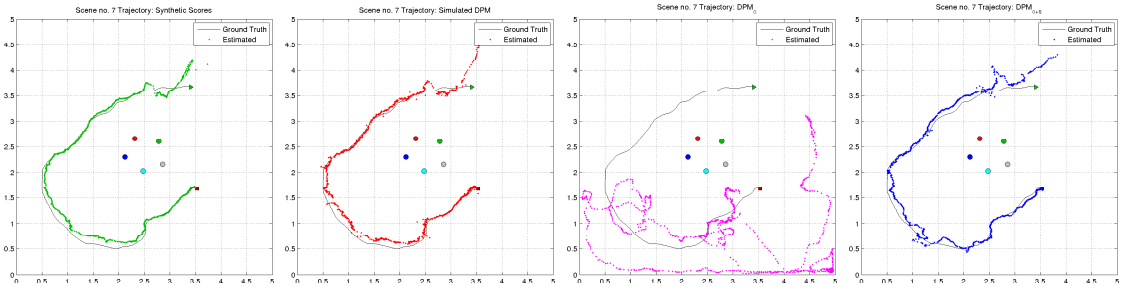


(e) Estimated trajectories for scene 5

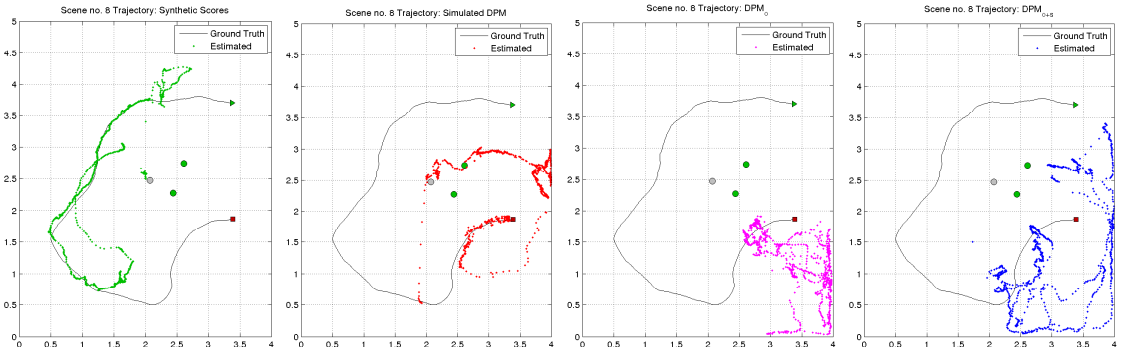


(f) Estimated trajectories for scene 6

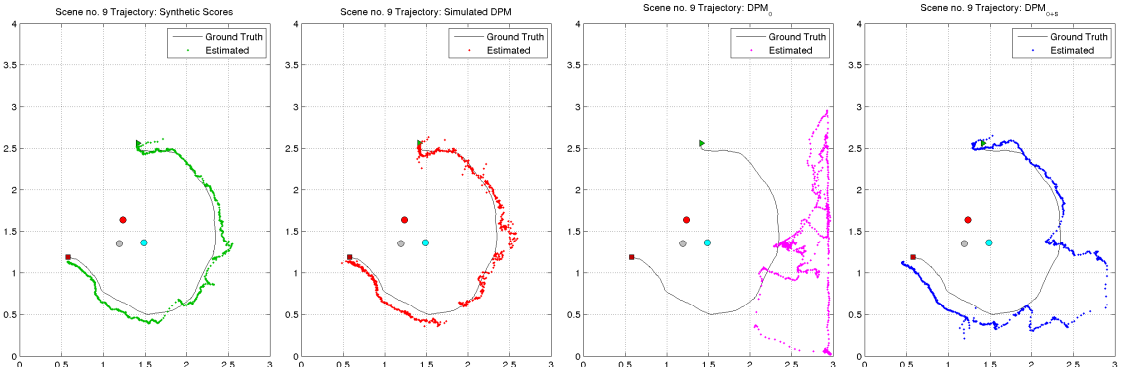
Figure 4.13: Estimated trajectories (continued). Position at each iteration computed from weighted average position of particles. From left to right: *synthetic-scores*, *simulated-DPM* , *objects*, *objects + scenes*.



(g) Estimated trajectories for scene 7

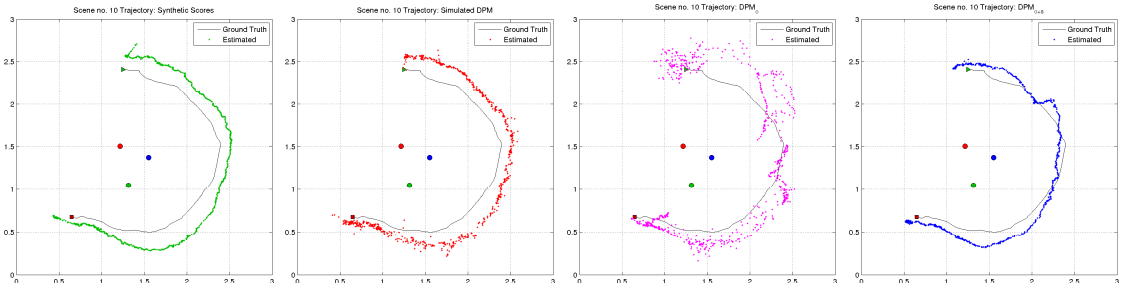


(h) Estimated trajectories for scene 8

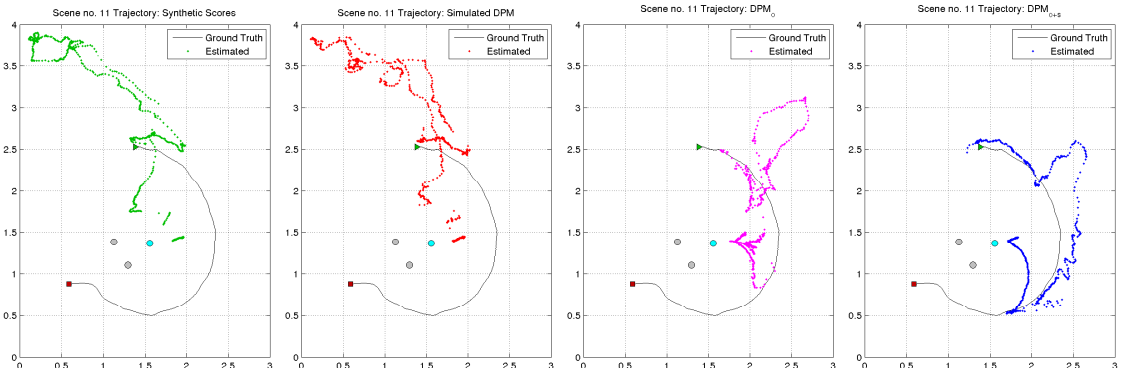


(i) Estimated trajectories for scene 9

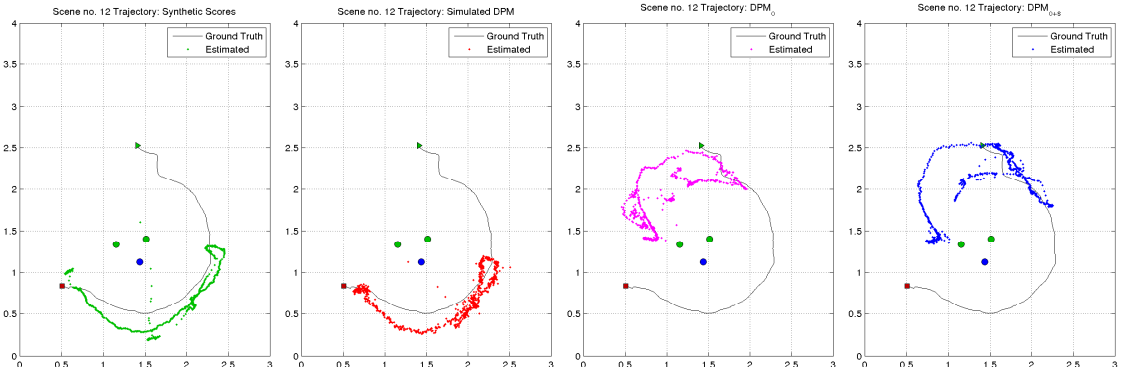
Figure 4.13: Estimated trajectories (continued). Position at each iteration computed from weighted average position of particles. From left to right: *synthetic-scores*, *simulated-DPM* , *objects*, *objects + scenes*.



(j) Estimated trajectories for scene 10



(k) Estimated trajectories for scene 11



(l) Estimated trajectories for scene 12

Figure 4.13: Estimated trajectories (continued). Position at each iteration computed from weighted average position of particles. From left to right: *synthetic-scores*, *simulated-DPM* , *objects*, *objects + scenes*.

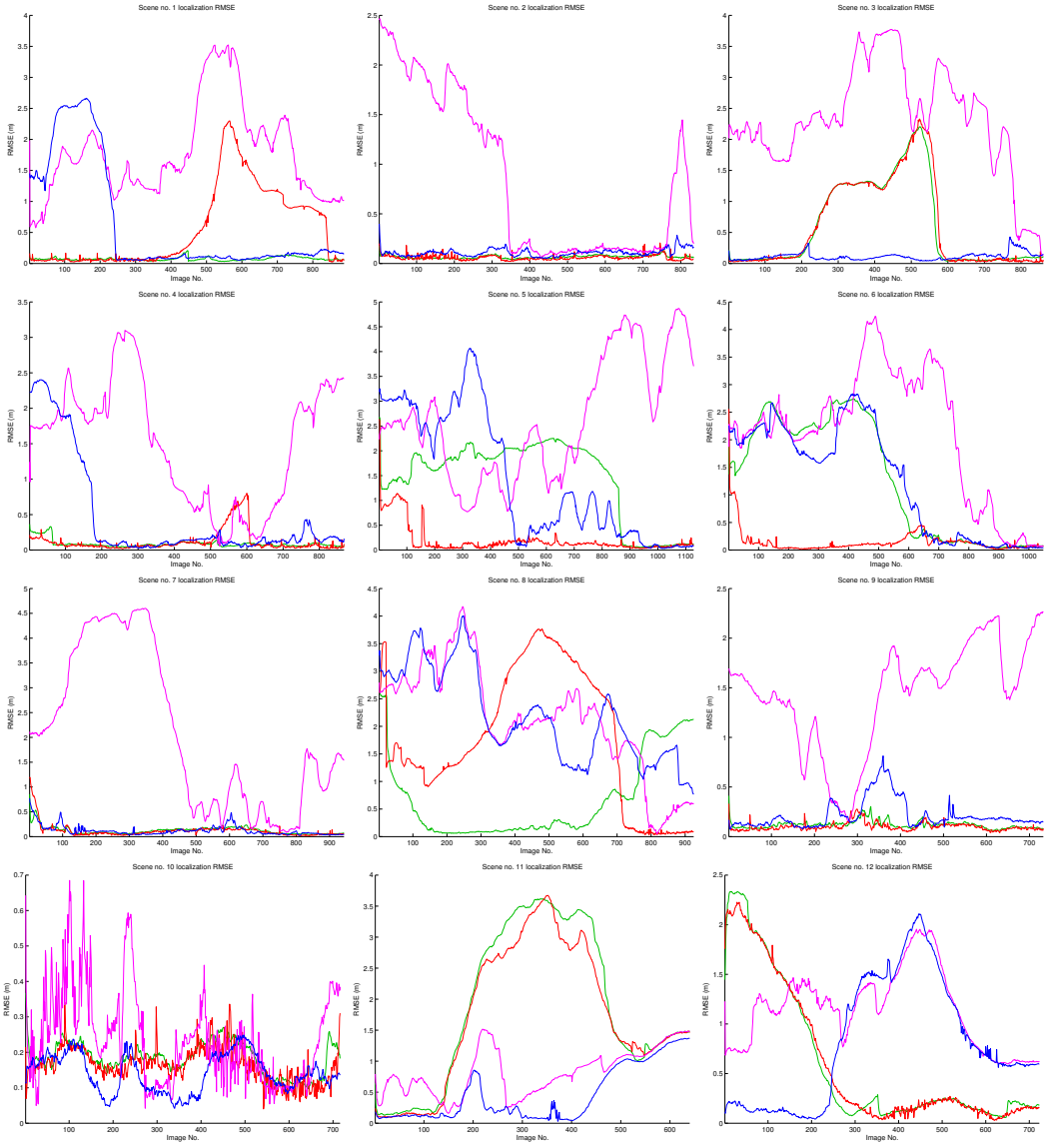


Figure 4.14: Localization error. Root-mean-square of the error between the recovered trajectory and the ground-truth trajectory. From left to right: *synthetic-scores*, *simulated-DPM* , *objects*, *objects + scenes*.

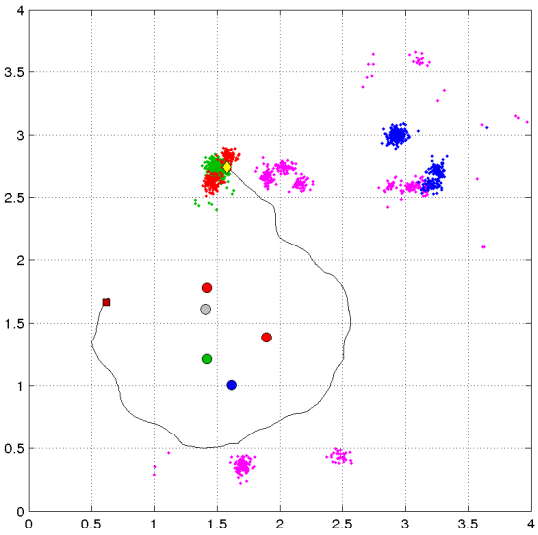
Detection Errors

Observation likelihoods (equation (4.28)) are used to weigh the particles during localization. These rely on the underlying object detector to provide scores that match the learned score distributions (section 4.8.2). When the detector fails, either by giving a low score to a positive detection, or a high score to a negative detection, it reduces the weight of otherwise likely particles. These kind of errors are responsible for the majority of localization failures for the DPM_o model (marked in purple).

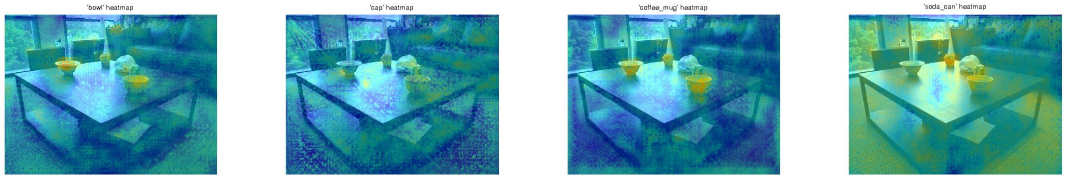
Examples of these kind of errors with the DPM_{o+s} model can be seen in scene 1, 3 and 4 (figures 4.13a, 4.13c and 4.13d). Early in scene 1 (figure 4.15d) both the *coffee_mug* and *soda_can* detectors contain high values on *bowl* and *coffee_mug* instances respectively. These score are very unlikely for negative detections in the learned distribution (figure 4.11). Once the detector scores fall within expected range, localization converges to the right area by frame 244 (figure 4.16). The same problem occurs in scene 4 (figure 4.18 and figure 4.19). Similarly, towards the end of scene 3 (figure 4.17d) a miss-identification of a *bowl* as a *coffee_mug* negatively affect localization for a short period. Since these errors are introduced during object detection, they do not affect the simulations.

More discriminative detectors would help reduce these problems. It's also possible to train 1-vs-1 detectors and adjust the model accordingly. Another possibility is explicitly modelling the detector cross-class confusion, discarding assumption 3 from section 4.7.2. This requires learning pairwise score likelihood distributions but could employ the existing DPM models.

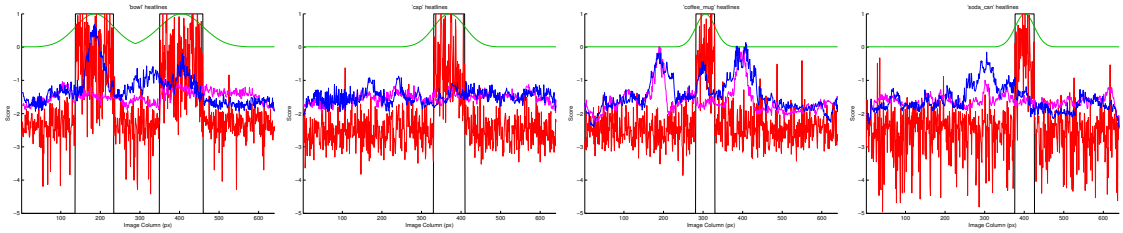
(a) Original image



(b) Particles



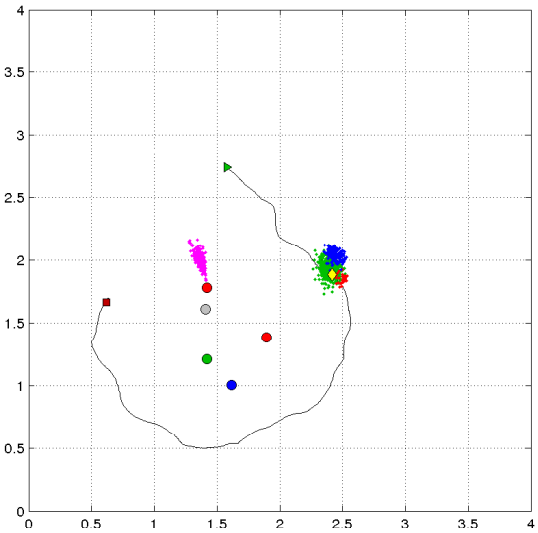
(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})



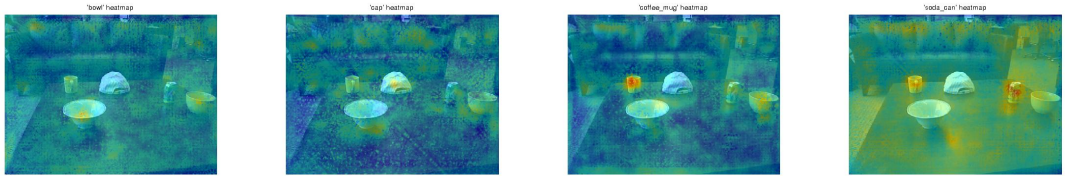
(d) Heatlines for all object classes

Figure 4.15: Particle filter and detector, scene 1 image 1. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: bowl, cap, coffee_mug and soda_can. Localization failure stemming from hallucinations (purple, blue).

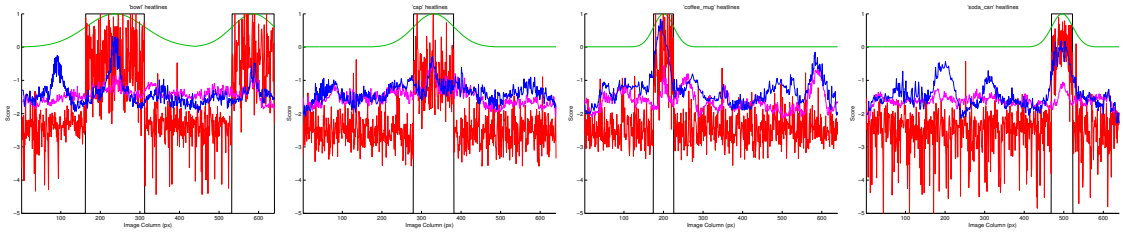
(a) Original image



(b) Particles



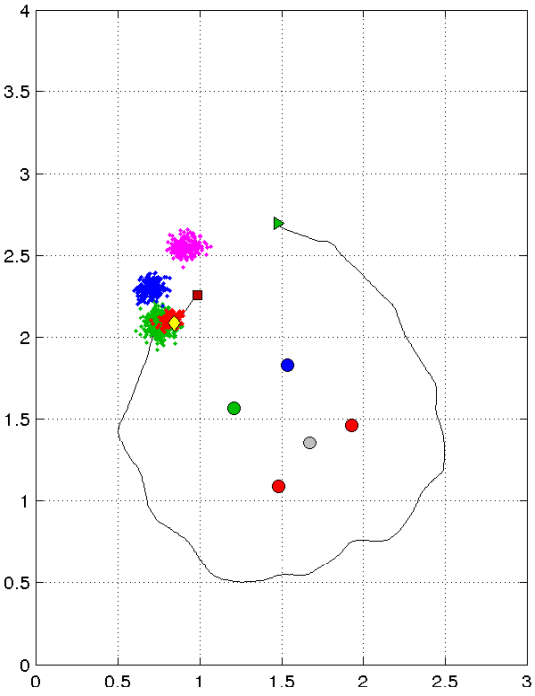
(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})



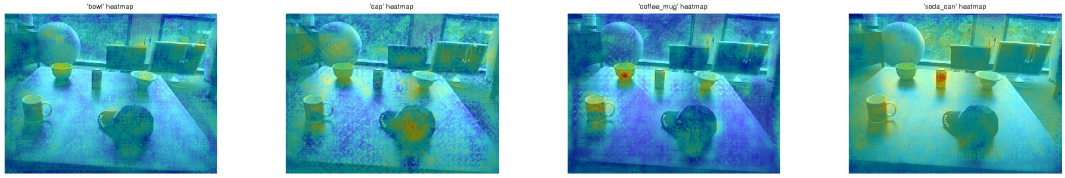
(d) Heatlines for all object classes

Figure 4.16: Particle filter and detector, scene 1 image 244. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: bowl, cap, coffee_mug and soda_can. Localization recovers (blue).

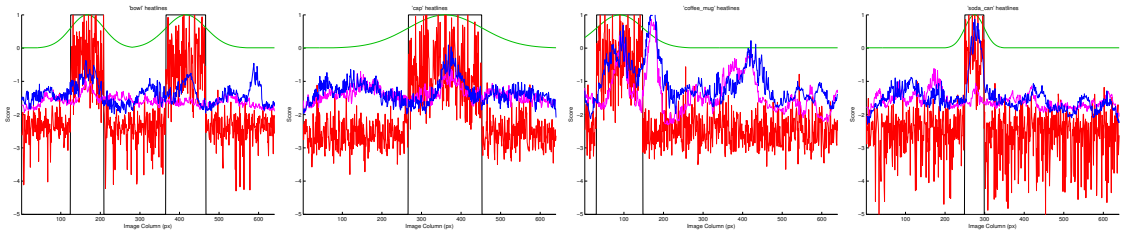
(a) Original image



(b) Particles



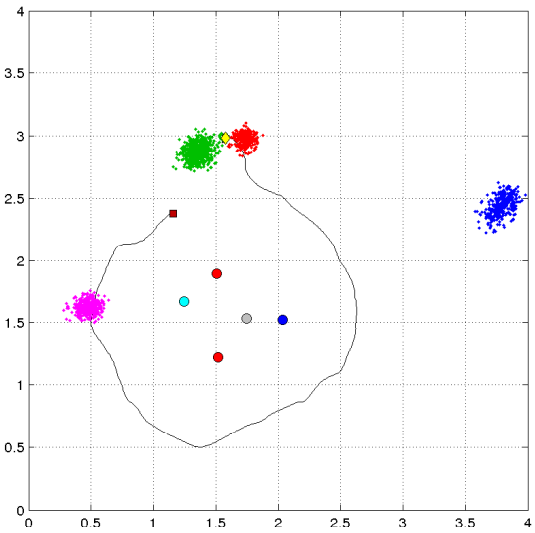
(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})



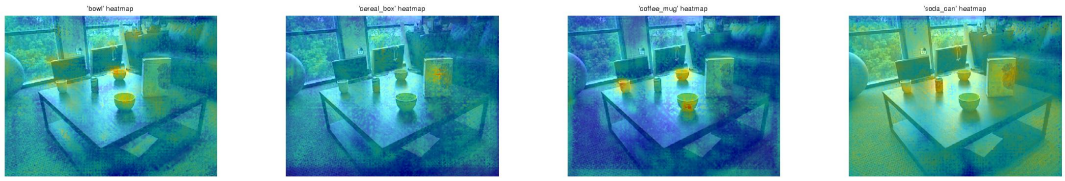
(d) Heatlines for all object classes

Figure 4.17: Particle filter and detector, scene 3 image 800. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: bowl, cap, coffee_mug and soda_can. Localization adversely affected by hallucinated object (purple, blue).

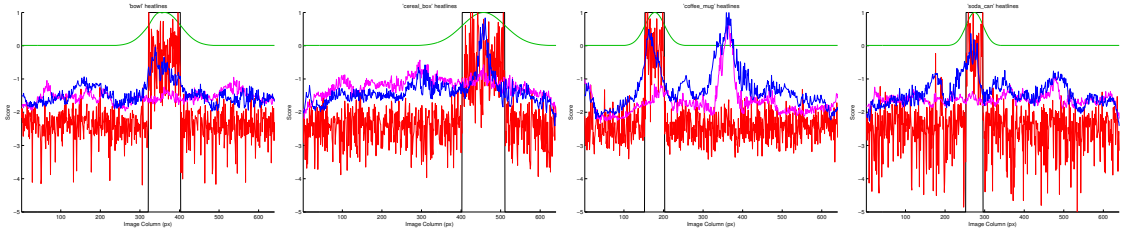
(a) Original image



(b) Particles



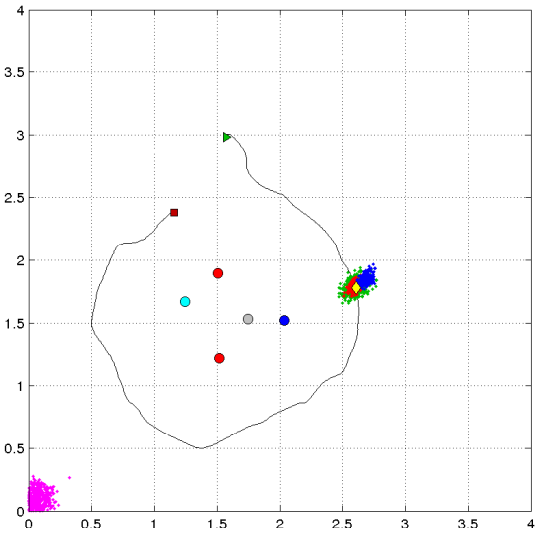
(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})



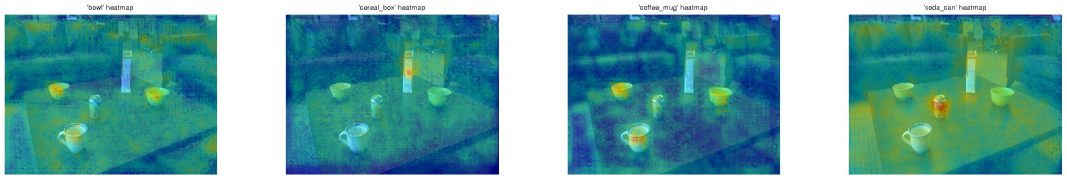
(d) Heatlines for all object classes

Figure 4.18: Particle filter and detector, scene 4 image 10. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: bowl, cereal_box, coffee_mug and soda_can. Localization failure stemming from hallucinations (purple, blue).

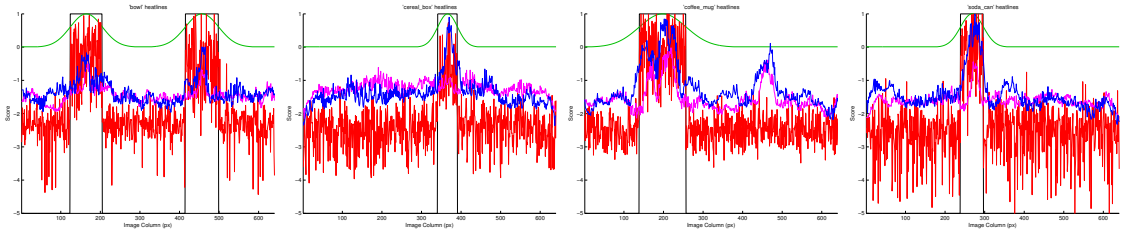
(a) Original image



(b) Particles



(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})



(d) Heatlines for all object classes

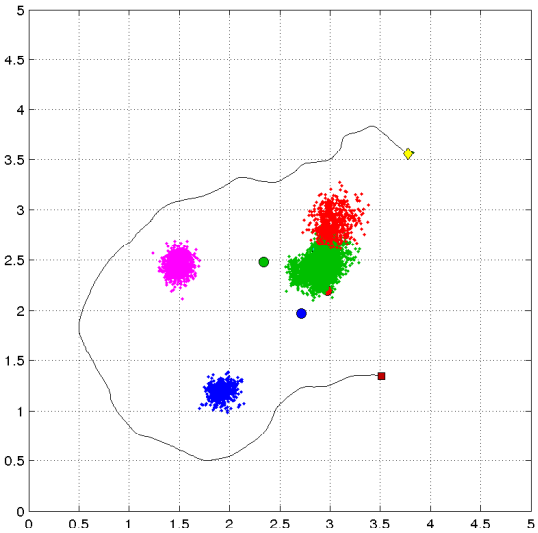
Figure 4.19: Particle filter and detector, scene 4 image 250. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: bowl, cereal_box, coffee_mug and soda_can. Localization recovers (blue).

Not Enough Objects

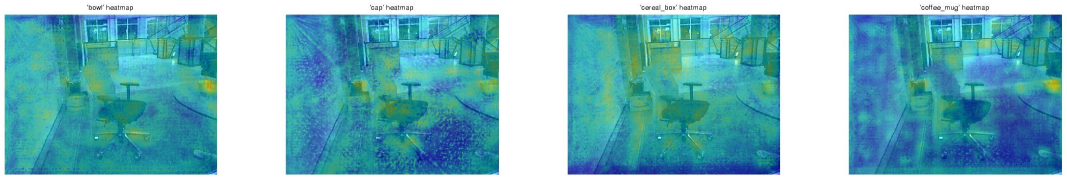
Several sequences include significant sections where few or none of the known objects are visible in the scene. In those cases localization falls prey to noise in the detector and hallucinations. These include scenes 5, 6 and 8 (figures 4.13e, 4.13f and 4.13h). For many frames in the beginning of scene 5, only one object is visible (*bowl*). Normally this would yield an ambiguous location, but combined with a *coffee_mug* hallucination (figure 4.20d) results in tighter particle cluster (figure 4.20b, blue vs. green). The particle filter favors locations where a *bowl* and *coffee_mug* lie approximately in a straight line from the camera. *Simulated-DPM* particles (red) are close enough to the true position that they approach the right solution when more object become visible while the other methods fail (figure 4.21). The same happens in scene 6 with the hallucination of a *soda_can* from a *coffee_mug* (figure 4.22b, blue, purple). Eventually both *simulated-scores* and DPM_{o+s} approach the right solution and track for the remainder of the scene (figure 4.22b), blue, green). Scene 8 is similar, but eventually *synthetic scores* localizes (figure 4.24b, green).

These errors affect both simulations and detector based approaches. They are inherent to the setup of the experiment, with many images captured without objects or similar objects aligning. One option to address these issues is to use a larger field of view camera, for example the Ladybug camera used in section 4.5. Larger fields of view would increase the number objects visible in an image, and help disambiguate when objects align together. Richer object maps would also assist for the same reasons.

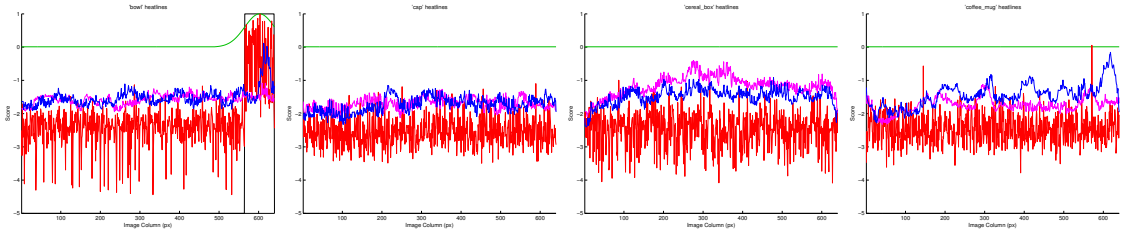
(a) Original image



(b) Particles

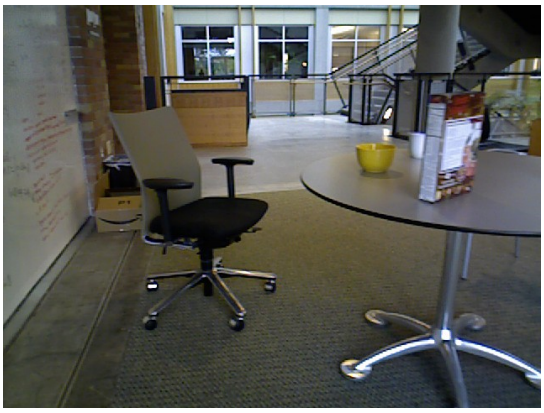


(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})

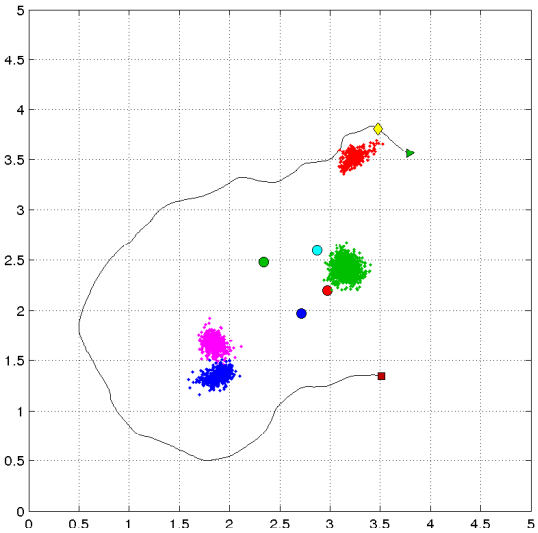


(d) Heatlines for all object classes

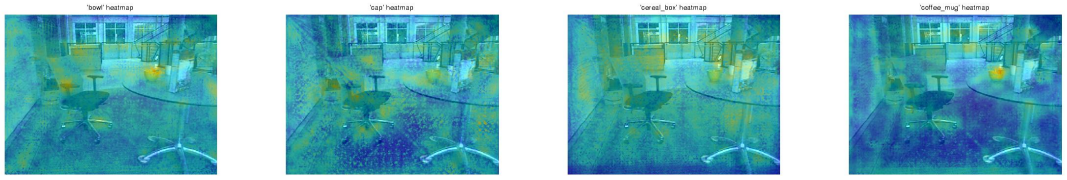
Figure 4.20: Particle filter and detector, scene 5 image 50. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: bowl, cap, cereal_box and coffee_mug. Localization failing for lack of objects.



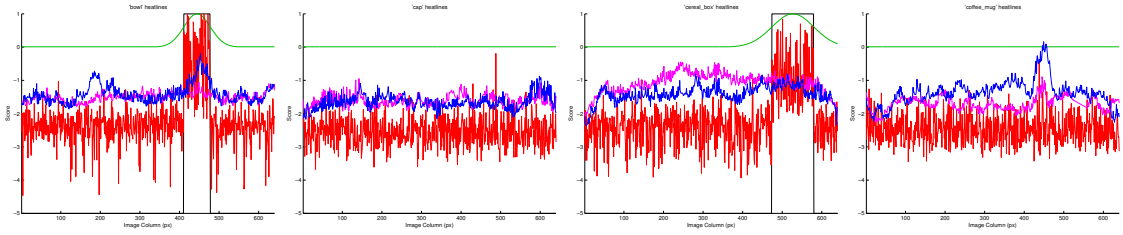
(a) Original image



(b) Particles

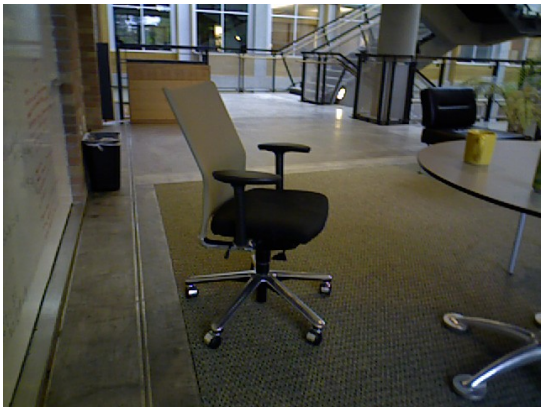


(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})

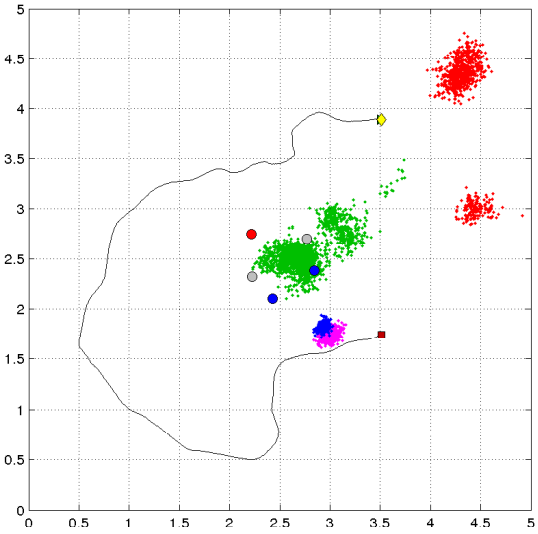


(d) Heatlines for all object classes

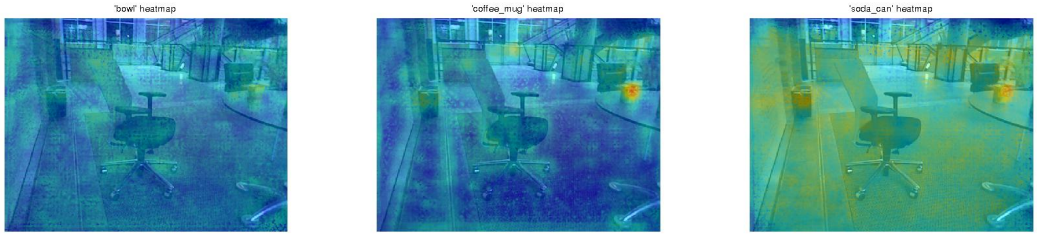
Figure 4.21: Particle filter and detector, scene 5 image 110. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: bowl, cap, cereal_box and coffee_mug. Localization recovering with additional objects (red).



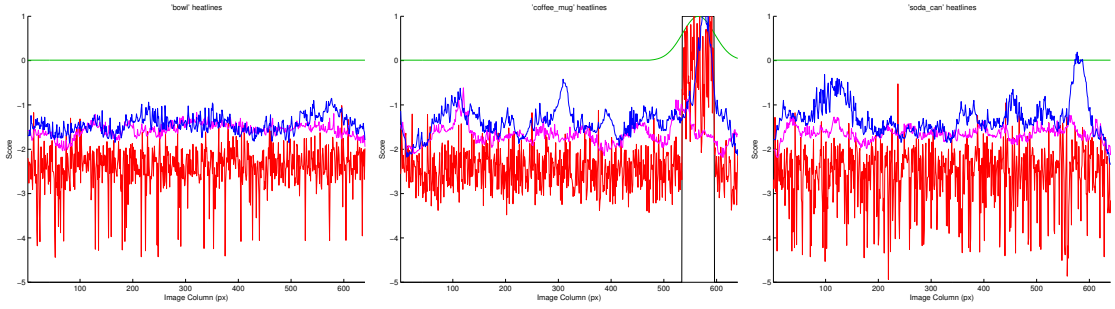
(a) Original image



(b) Particles



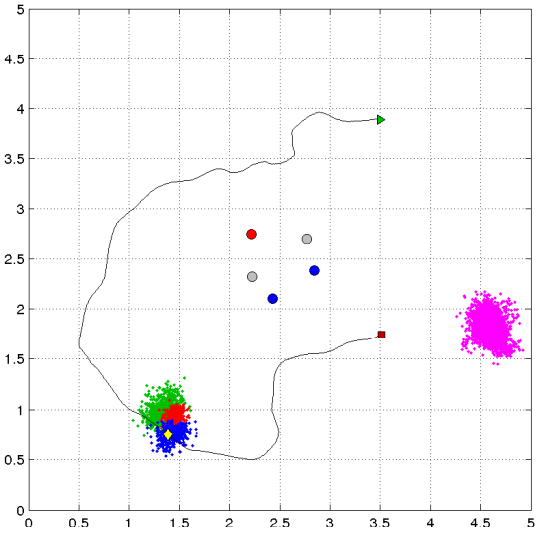
(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})



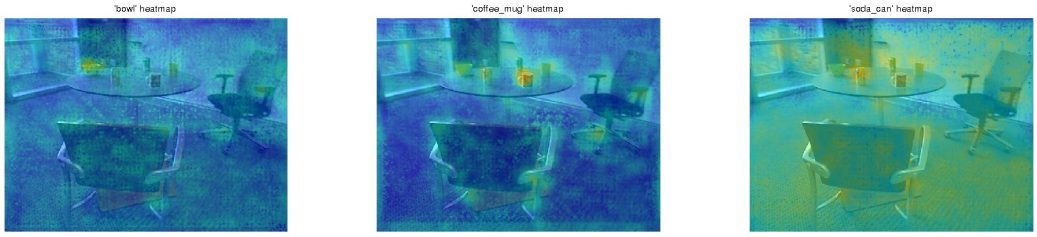
(d) Heatlines for all object classes

Figure 4.22: Particle filter and detector, scene 6 image 10. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: bowl, coffee_mug, soda_can. Localization failing for lack of objects.

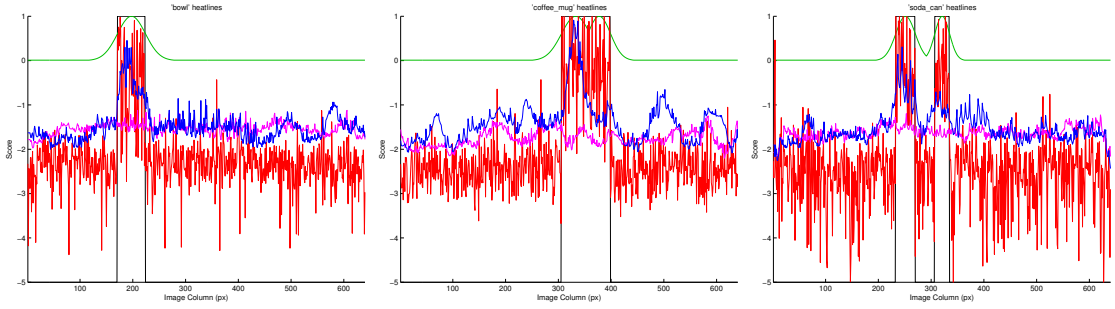
(a) Original image



(b) Particles

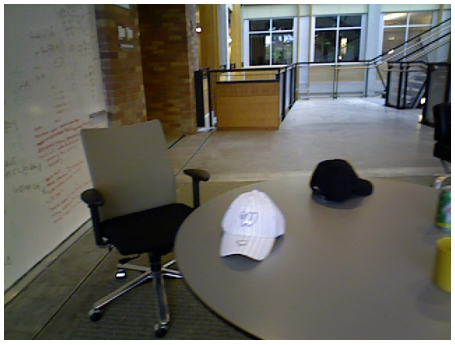


(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})

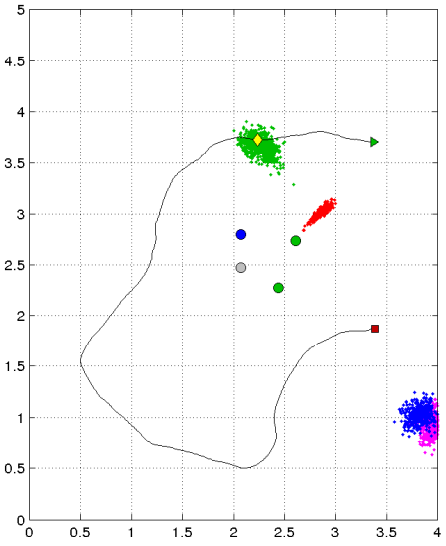


(d) Heatlines for all object classes

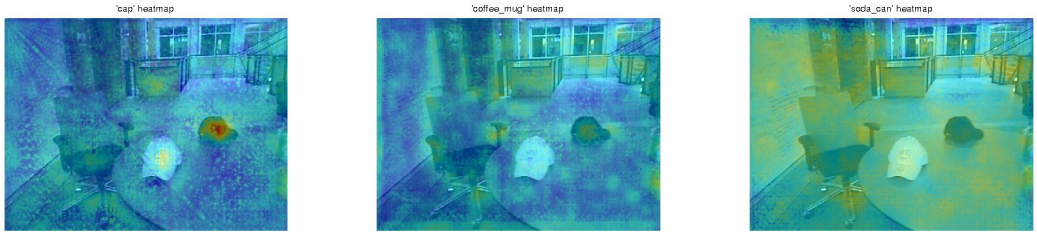
Figure 4.23: Particle filter and detector, scene 6 image 10. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: bowl, coffee_mug, soda_can. Localization recovers over time (green, blue).



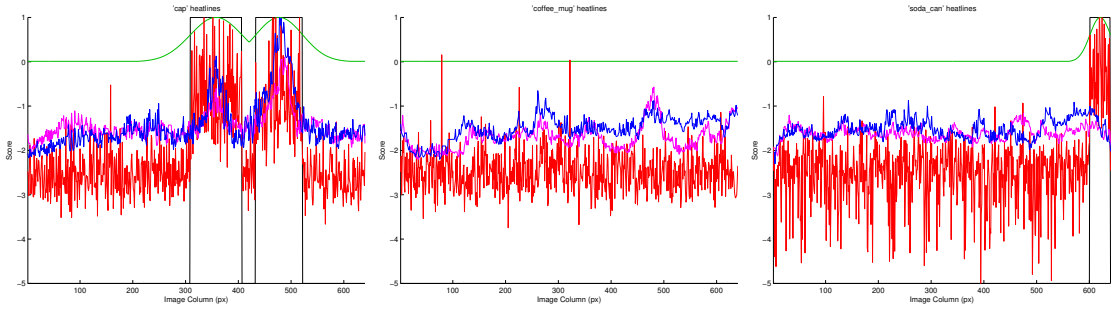
(a) Original image



(b) Particles



(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})



(d) Heatlines for all object classes

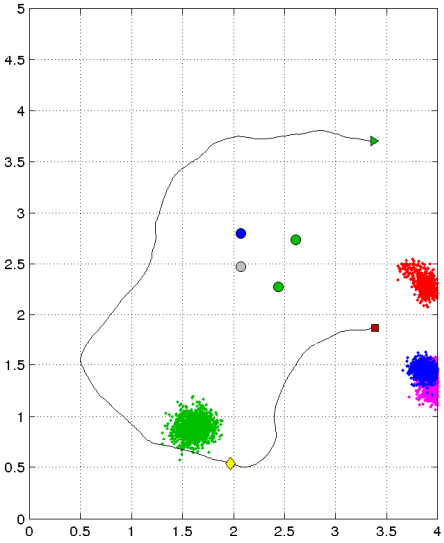
Figure 4.24: Particle filter and detector, scene 8 image 150. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: cap, coffee_mug, soda_can. Localization failing for lack of objects (red, purple, blue).

Occlusions

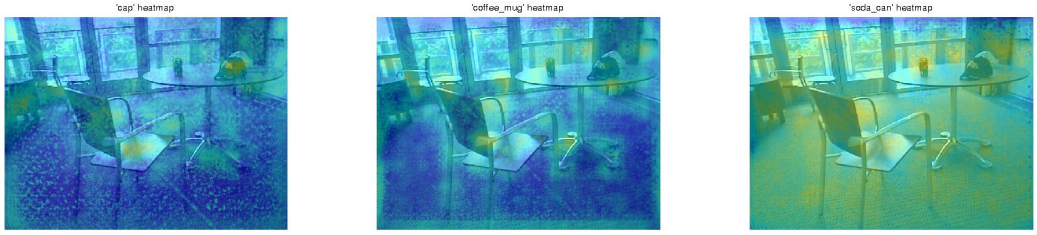
Although similar to not having enough objects, occlusions are different. Before, the lack of objects in the image was modelled correctly. Although sparse, that information can still inform localization. If no objects are visible, then particles that aren't expected to see objects will get higher weights. With occlusions objects are not visible, but are still expected. This is a shortcoming of the detection model in equation (4.25). It can be expanded to explicitly discount objects that should not be visible even though they're in the field of view. This requires modelling occlusions between objects, measuring object sizes and distances and mapping non-object occluding surfaces (e.g. walls).

Localization error due to occlusions can be seen in scenes 8, 9 and 11 (figures 4.13h, 4.13i and 4.13k). In scene 8, frame 650 the *coffee_mug* is hidden by the *soda_can* and one *cap* is hidden by other. With only two visible objects the particles disperses (figure 4.25b, green). The particle filter successfully localizes during the first 230 frames of scene 9 until the *soda_can* is occluded by the *cereal_box*. With only two visible objects the particles have more freedom of movement (figure 4.26b). This is exacerbated by frame 350 (figure 4.27) when the *bowl* is also hidden. With only one visible object there are no constraints on the particles. Once the *soda_can* reappears around frame 370 the particles converge on the correct location (figure 4.28 green, red, blue). Similar behaviour can be observed in scene 11, where localization diverges around frame 135 (figure 4.29).

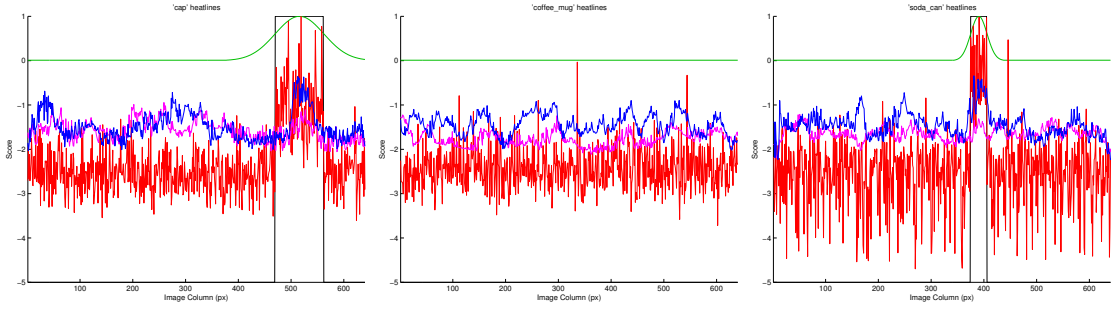
(a) Original image



(b) Particles



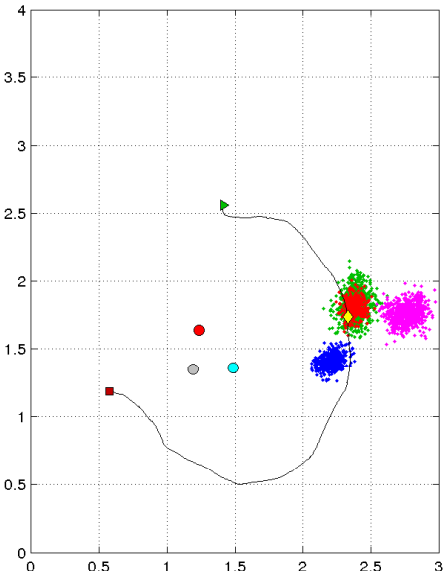
(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})



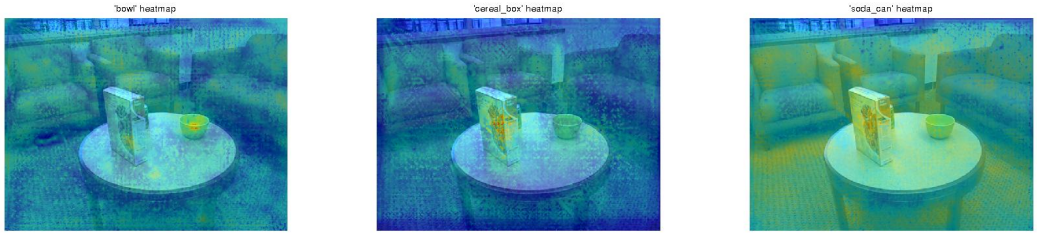
(d) Heatlines for all object classes

Figure 4.25: Particle filter and detector, scene 8 image 650. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: cap, coffee_mug, soda_can. Localization failing due to occlusions.

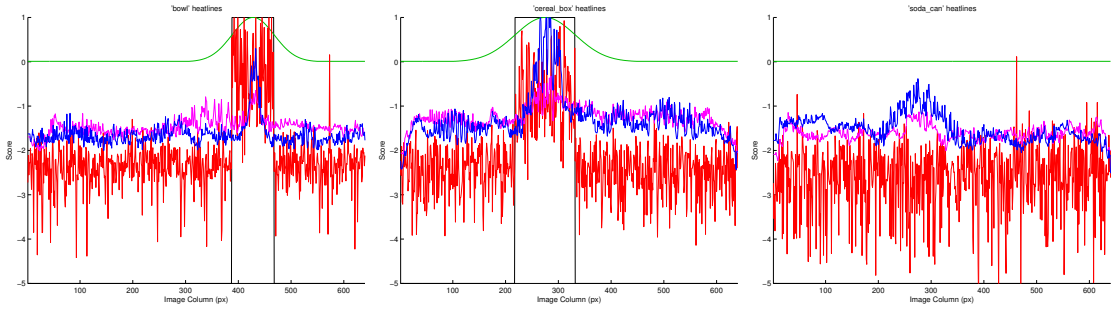
(a) Original image



(b) Particles



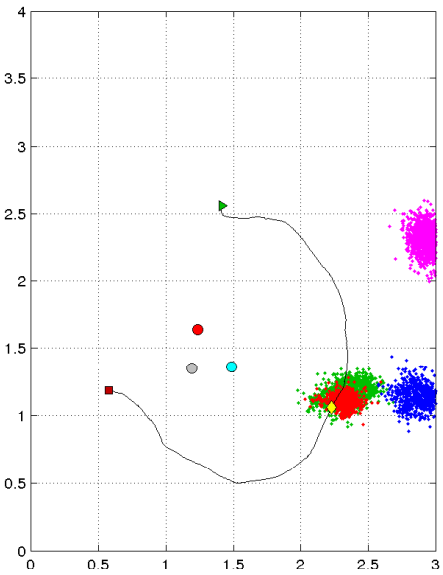
(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})



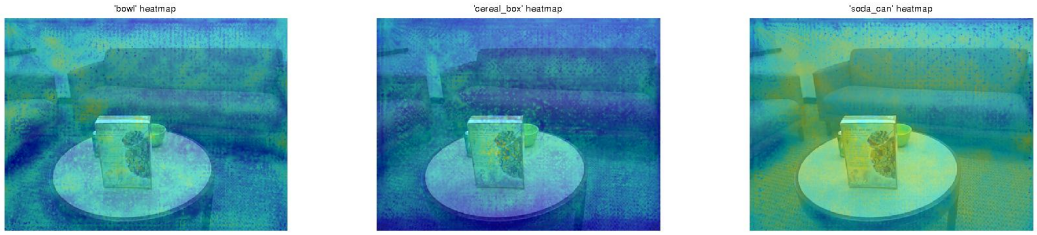
(d) Heatlines for all object classes

Figure 4.26: Particle filter and detector, scene 9 image 235. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: bowl, cereal_box, soda_can. Localization dispersing due to first occlusion.

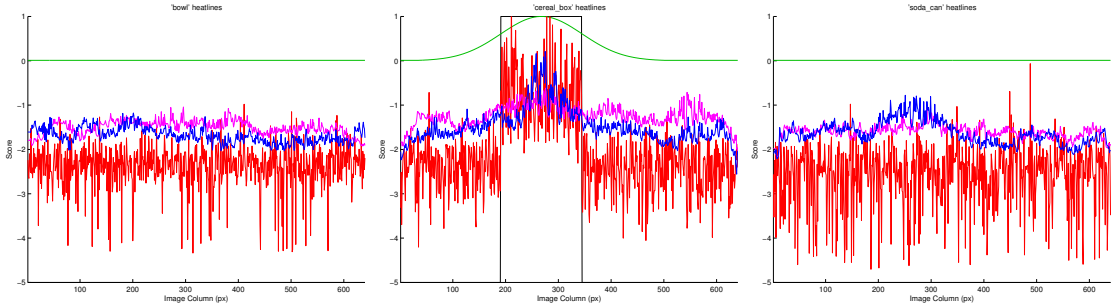
(a) Original image



(b) Particles



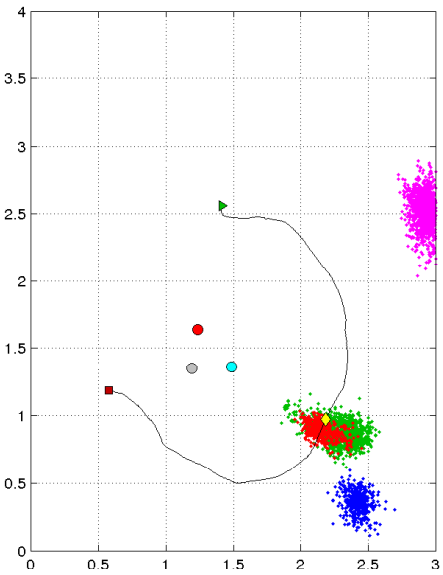
(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})



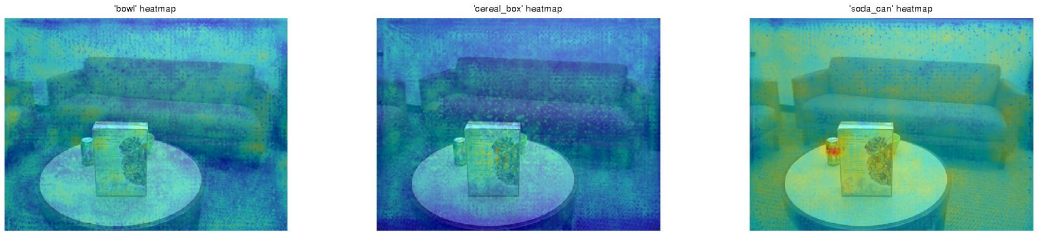
(d) Heatlines for all object classes

Figure 4.27: Particle filter and detector, scene 9 image 350. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: bowl, cereal_box, soda_can. Localization dispersing due to second occlusion.

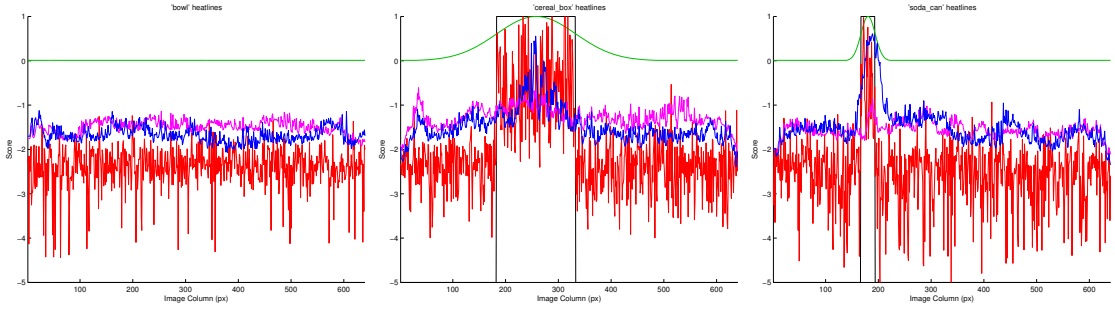
(a) Original image



(b) Particles



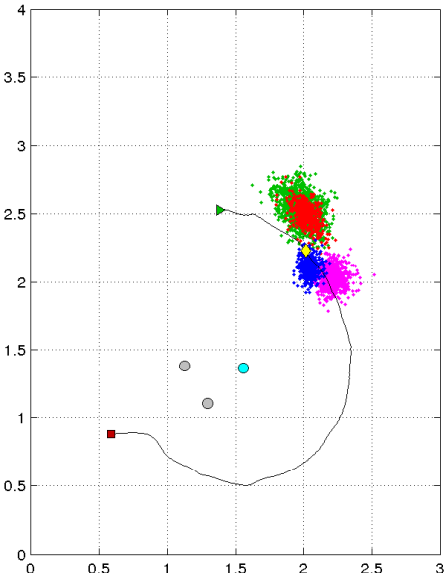
(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})



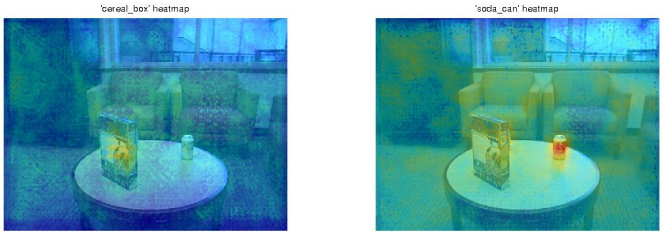
(d) Heatlines for all object classes

Figure 4.28: Particle filter and detector, scene 9 image 370. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: bowl, cereal_box, soda_can. Localization converges as objects reappear.

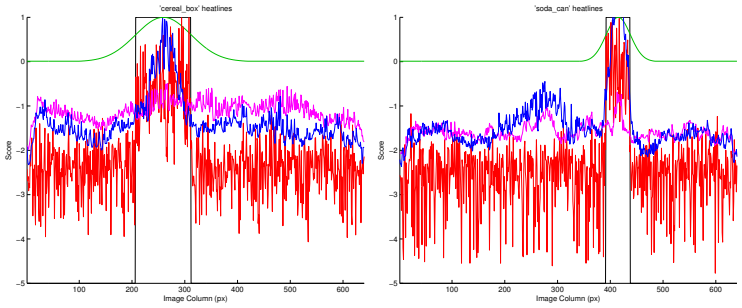
(a) Original image



(b) Particles



(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})



(d) Heatlines for all object classes

Figure 4.29: Particle filter and detector, scene 11 image 135. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: cereal_box, soda_can. Localization failing due to occlusions.

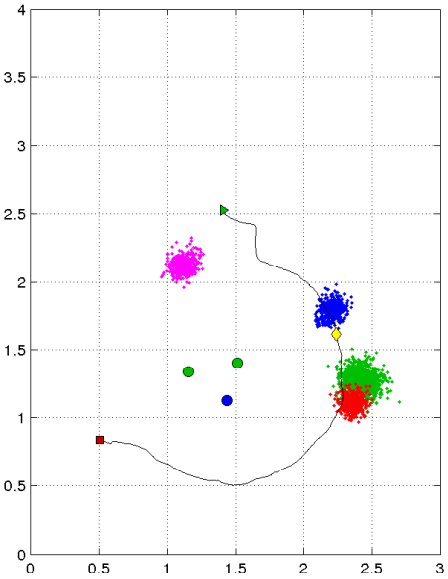
Geometric Ambiguity

Scene 12, on the other hand, suffers from geometric ambiguity. Although all three objects are visible, the two *cap* instances are aligned with the camera around frame 240 (figure 4.30, blue). This creates problems for the DPM_{o+s} localization[4]. Similarly, even though all three objects are visible, neither simulation converges until frame 275 (figure 4.31, green, red). At that point all three objects are visible and the two *cap* are starting to show angular divergence with respect to the camera.

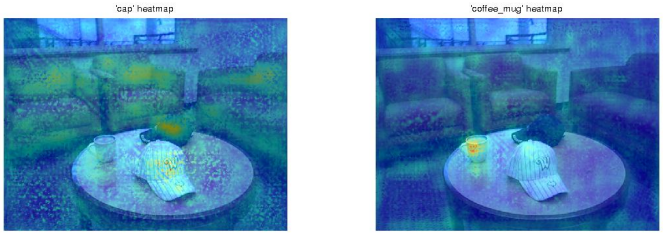
Geometric ambiguity exists in other scenes (e.g. scenes 7 and 11) but these are created by smaller / farther objects. Since the *cap* objects are larger, their superposition lasts longer and covers a greater section of the camera field of view. The ambiguity therefore lasts for more iterations, causing the particles to spread. These problems can be addressed in the same way as occlusions, by modelling object sizes and distances.

[4]Both simulations fail at the start of scene 12 as no ground-truth bounding boxes exist for one of the three objects until later in the sequence.

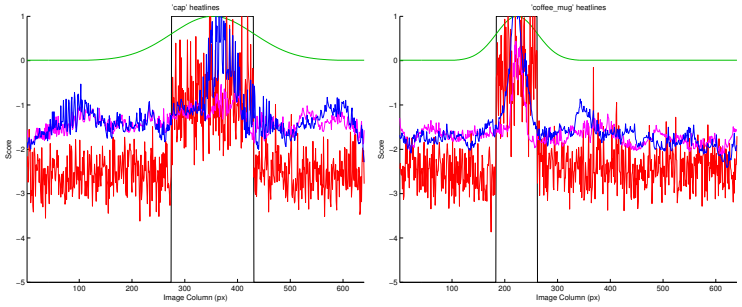
(a) Original image



(b) Particles



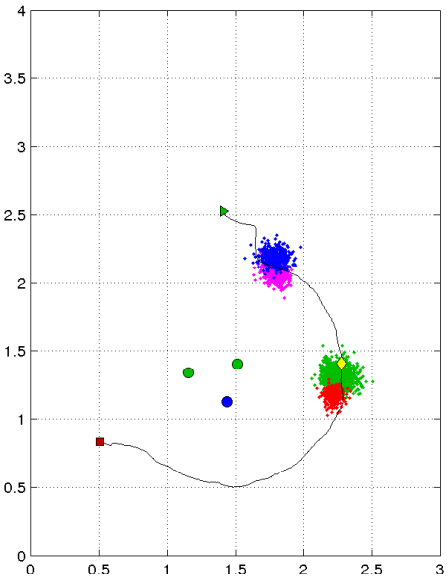
(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})



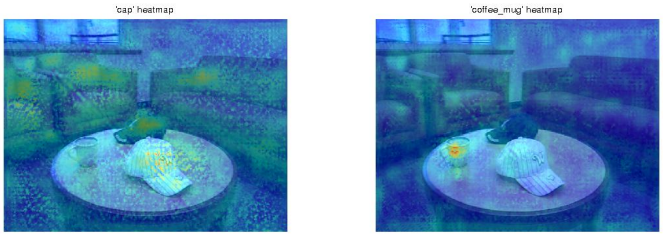
(d) Heatlines for all object classes

Figure 4.30: Particle filter and detector, scene 12 image 240. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: cap, coffee_mug. Localization failing due to *model* occlusions.

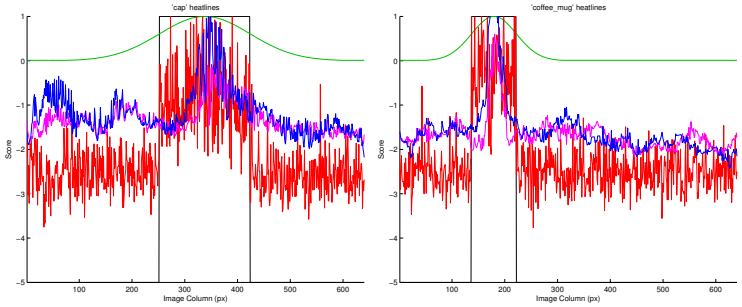
(a) Original image



(b) Particles



(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})

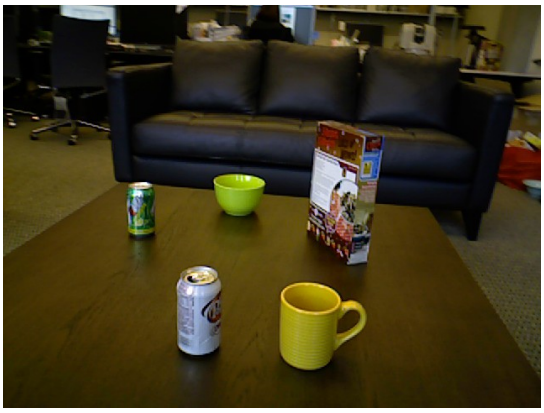


(d) Heatlines for all object classes

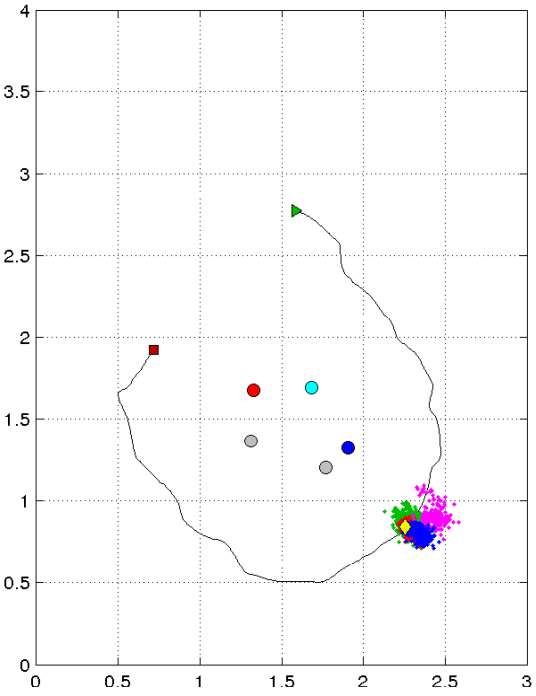
Figure 4.31: Particle filter and detector, scene 12 image 275. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: cap, coffee_mug. Localization succeeds after *model* occlusions end.

Successful Localization

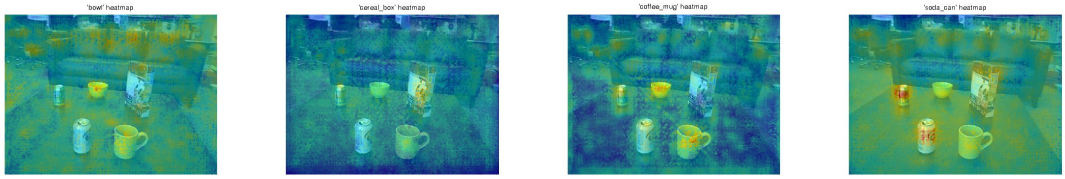
The particles converge closely to the ground truth position for scenes 2, 7 and 10 (figures 4.13b, 4.13g and 4.13j). These contain multiple objects and object classes in a spacious configuration, avoiding ambiguities. The detectors behaves well and doesn't assign unlikely scores to false detections and there are few shot-lived occlusions. The particle filter never accumulates more than 0.5 meters of error (section 4.8.4). Sample iterations are shown for each scene: 2 frame 400 (figure 4.32), 7 frame 100 (figure 4.33) and 10 frame 600 (figure 4.34).



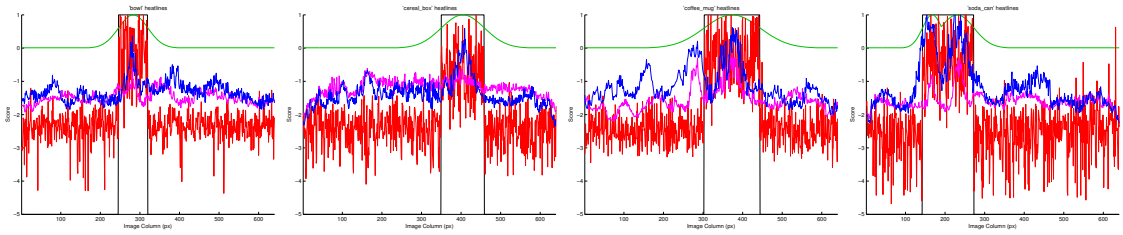
(a) Original image



(b) Particles

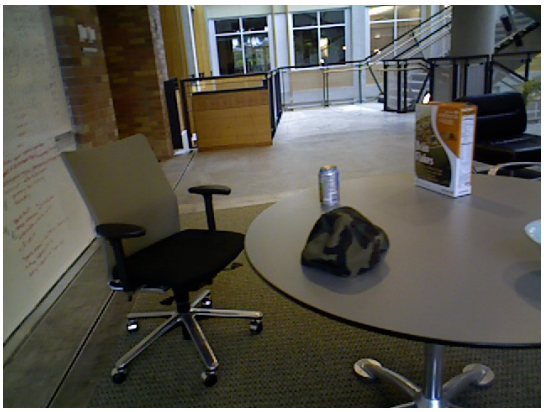


(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})

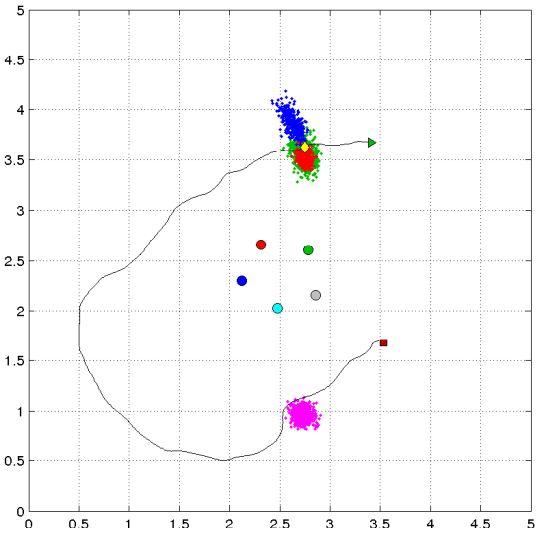


(d) Heatlines for all object classes

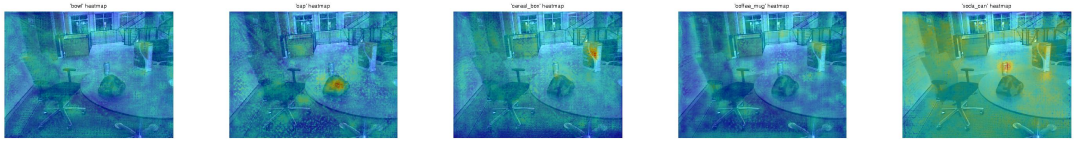
Figure 4.32: Particle filter and detector, scene 2 image 400. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: bowl, cereal_box, coffee_mug and soda_can. Successful localization.



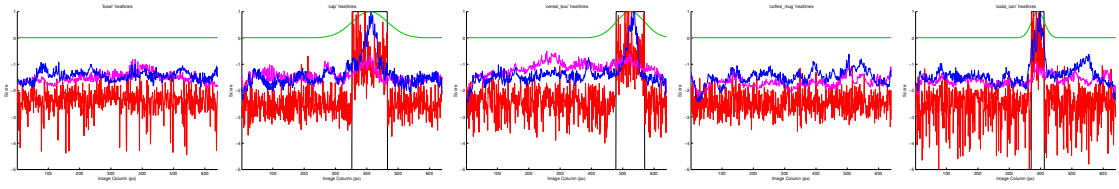
(a) Original image



(b) Particles



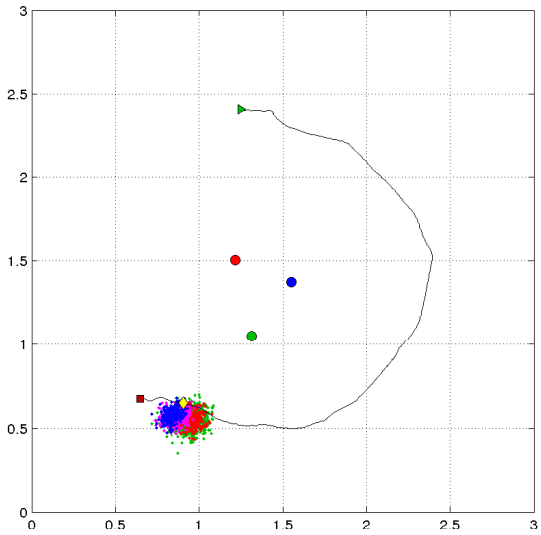
(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})



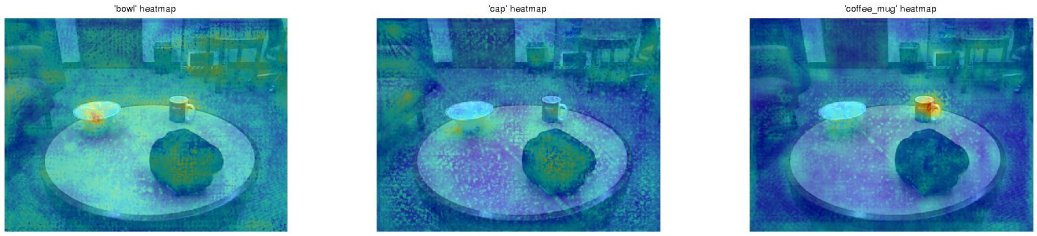
(d) Heatlines for all object classes

Figure 4.33: Particle filter and detector, scene 7 image 100. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: bowl, cap, cereal_box, coffee_mug and soda_can. Successful localization (green, red, blue).

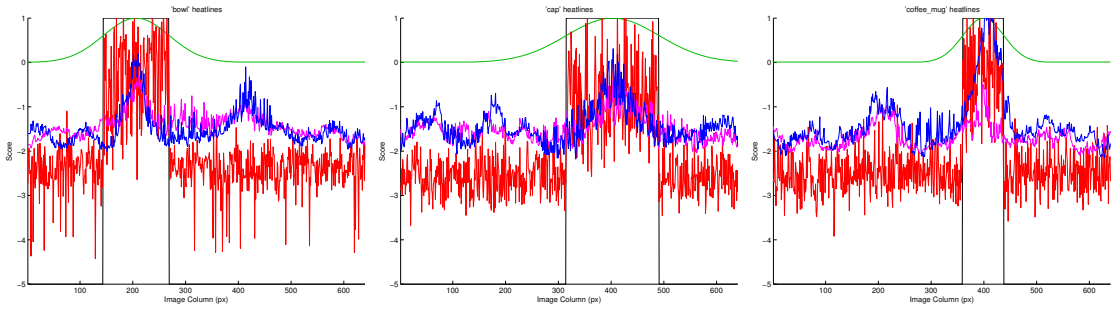
(a) Original image



(b) Particles



(c) Heatmap overlays for *objects* + *scenes* trained DPM (DPM_{o+s})



(d) Heatlines for all object classes

Figure 4.34: Particle filter and detector, scene 10 image 600. Colors for (b)(d): *synthetic-scores*, *simulated-DPM*, DPM_o, DPM_{o+s} and *ground-truth* (black). Object classes for (c)(d) from left to right: bowl, cap, coffee_mug and soda_can. Successful localization.

4.9 Summary

The proposed approach shows that it is possible to localize in challenging indoor environments using only objects from images in conjunction with a semantic map. During operation the system uses only visual sensors in combination with a rough representation of the environment. This rough representation is cheap and easy to construct either by hand or semi-automatically. Unlike Pronobis et al. [2010a,c] and Rottmann et al. [2005] which localize via classification, here true localization is performed. In fact, in the scenario presented here, classification based approaches would be completely useless as only a single room is explored. Semantics are directly incorporated by using object detections as features for localization. Instead of tackling the object detection problem as a separate component, it is instead tailored to the task of localization. The resulting object detection scheme relies on soft detections using object response maps, or heatmaps.

A particle filter is used to localize within the environment. In section 4.5 localization is achieved by comparing the observed heatmaps with the expected heatmaps generated from the annotated map. Whereas in section 4.8 object detection was explicitly modelled. By using the detected objects directly as landmarks, perceptual variability is decoupled from the localization problem. Once an object is detected, its appearance has no impact on localization. The particles conforming to the map receive higher weights, clustering together and localizing the robot. Integrating landmarks over multiple frames yields more robust localization. Even if two separate locations have similar object configurations, combining information from previous (or subsequent) frames can distinguish the two locations.

Object detection in this situation is difficult due to motion blur and robot motion, both diverging from assumptions made by standard object detection approaches. These challenges were addressed by using a dense representation that incorporates some flexibility to illumination and motion variation. First, soft-object detection was carried out using two properties of images to generate heatmaps: gradient energies and colors. Although a richer library of image features could increase the detector's discriminability, discriminantly trained deformable parts models (DPM) are used instead. DPMs, a state-of-the-art

object detector was adapted for use in localization. Generating heatmaps from traditional bounding boxes allowed incorporation of weak signals that would otherwise be ignored. Although requiring substantial training data and training time, DPM based heatmaps were successfully used for localization.

In addition to incorporating more image features or different object detectors, additional information can be incorporated to improve detections. These include object pose, tracking across frames and more semantic reasoning (e.g. contact surfaces, occlusions, etc.). Object sizes and distances also aren't used, these would result in more accurate observation likelihoods and detection likelihoods. Additionally, temporal information is not used in the detection stage. Devising a method to "track" object heatmaps through time (as opposed to tracking objects) would also improve heatmap quality. Inter-class confusion could also be incorporated explicitly, allowing object detector i to inform the likelihood of the score for an object of class j. This would allow less discriminative object detectors at the expense of a more complex observation model.

Finally, the need for a semantic map already annotated with localized objects is a significant burden on users of this system. It's assumed that the map is static, and that objects do not move. Any updates to the environment would have to be adjusted manually. Incorporating a traditional mapping solution is insufficient as it lacks automatic localization of semantically meaningful objects. In this chapter, a semantic map was generated in a semi-automatic fashion using semantically labelled 3D point clouds from Lai et al. [2014]. Cleveland et al. [2015] also present a system for creating semantic maps that could be used with this localization framework. A semantic mapping solution is presented in chapter 5 incorporating soft detections to fulfill this requirement. Soft detection, unlike traditional object detection frameworks, is built specifically to tackle the challenges of localization and associated imagery.

Chapter 5

Semantic Mapping with Object Heatmaps

5.1 Introduction

The state of the art in robotic localization and mapping is moving beyond metric representations in terms of point clouds to semantic descriptions. Scenes are not marked by geometric qualities such as Euclidean coordinates but are rather labeled with meaningful keywords (e.g. kitchen and office) or by their contents (e.g. refrigerator, oven, printer and monitor). This transition from a low-level signal like (XYZ,RGB) tuples to a map of symbols is based on the robot's ability to successfully recognize objects, scenes, and represent their relations. Although single picture object recognition has made significant progress, current precision-recall performance does not allow for safe navigation and manipulation of the environment by robots. Nevertheless, robots like biological species are not single-picture processors. They have access to multiple modalities (RGB, depth, audio) and they can acquire multiple views.

Chapter 4 introduced a method of localizing in a large environment through the use of objects. The position and classification of objects is known apriori, provided via a

semantincally annotated map. Such maps could be sketched by hand, or generated automatically through a separate process.

In this chapter, a purely vision-based approach is presented to solve the problem of 3D semantic mapping. It is assumed camera pose is given from some visual odometry algorithm, in this case RGB-D SLAM [Engelhard et al., 2011; Henry et al., 2010]. A 3D model of the object is acquired using a Kinect camera and used to render training image exemplars from multiple views around an object. Using the Dominant Orientation Template (DOT) [Hinterstoisser et al., 2010] detector a score for each pixel in the image. Per-pixel scores define heatmaps corresponding to each object and each of the view exemplars. Given the camera pose, image heatmaps are back-project to create a world heat-volume. Each point in space is now associated with a confidence score that an object is there. When the camera moves, a new image heatmap is obtained, it is back-projected into the world heat map and the world heat-volume is updated. The maximum over all views and objects in space can be chosen at any point, "freezing" the estimation process to produce a map with objects. The point-cloud data is never used after recovering object pose.

The primary contribution is a novel vision-based approach for constructing semantic maps from given camera pose and very simple object exemplar images. An object response score is calculated for each pixel, which is then projected into 3D space. Responses across multiple frames are integrated to find and determine the locations of objects in the scene. The approach has several advantages: it is incremental, fast and avoids hard decisions by integrating dense detector responses. Instead of learning an object class the system relies on detecting the same instance of an object. This does not change the principle of the approach, as all it requires is a dense similarity responses from an object detector across every pixel in the image.

Next, related work is reviewed in section 5.2. Then the proposed system is presented in its two parts. First, section 5.3 reviews the object detection scheme from section 4.3 which generates object heatmaps. Then section 5.4 covert how heatmaps are aggreagted

over space to create volumes, and how final composition of the scene is recovered. Results are presented in section 5.5, concluding with a summary in section 5.6.

5.2 Related Work

Semantic mapping approaches focus on two main challenges: labeling the substrate of the scene (ground plane, walls, etc) and recognizing objects. The former is related to the world modeling literature [Burgard and Hebert, 2008] and in particular to terrain classification and place labeling. Wolf and Sukhatme [2008] label 3D maps based on traversability of terrain using hidden Markov model (HMM) and support vector machine (SVM) techniques. A representative of the state of the art approach in place labeling can be found in Pronobis et al. [2010a]. A significant boost to the state of the art in processing and labeling point clouds has been given by the introduction of the PCL library [Rusu and Cousins, 2011].

This chapter concentrates on object-based mapping and, thus, mainly refers to these in this overview. Chapter 4 used soft detection representations to detect objects whose position is known in a prior map. In this chapter, objects are detected and localized using view exemplars. With respect to Kuipers [2000] spatial semantic hierarchy, this approach is concerned with the naming of entities in the sensorial levels of the pyramid although it required a traversal of the environment in order to accomplish this.

Among the approaches which label objects it is appropriate to differentiate between approaches which first build a 3D map and then label the 3D points/surfaces [Anguelov et al., 2002; Blodow et al., 2011; Civera et al., 2011; Gunther et al., 2013; Nüchter and Hertzberg, 2008; Oberlander et al., 2008; Salas-Moreno et al., 2013; Trevor et al., 2010; Wong et al., 2013, 2015] versus approaches, like the one presented here, which directly build a map consisting of the objects [Atanasov et al., 2013; Bao et al., 2012, 2010; Cleveland et al., 2015; Ekvall et al., 2007; Limketkai et al., 2005; Sengupta et al., 2013]. The

significant differences between these approaches is that the former needs a 3D reconstruction of the objects while the latter can extract labels from RGB or range images.

Anguelov et al. [2002] detect non-stationary objects in occupancy maps and learn shape templates from them. Nüchter and Hertzberg [2008] established point clouds by following the projection of occluding contours on virtual views where humans are recognized using shape-based learning. Rusu et al. [2008] and Civera et al. [2011] apply appearance and geometry based recognition to annotate feature maps established with monocular SLAM. In Rusu et al. [2008], point clouds are first segmented into geometric shapes and objects are detected through model fitting techniques. Similar annotation of features or regions on top of SLAM are undertaken in Oberlander et al. [2008]; Trevor et al. [2010]. Gunther et al. [2013] on the other hand, derive surface meshes from point clouds, and match them to object CAD models. Hinterstoisser et al. [2012] use CAD based RGBD templates to match objects in RGBD images. On the other hand, Salas-Moreno et al. [2013] recover and 3D meshes as templates.

Wong et al. [2013, 2015] first recover a supporting surface, then use that to segment different point cloud clusters into separate objects. Like the work presented here, they explicitly integrate partial object information over time. However, they explicitly recover distinct object instances and cluster them, making their approach susceptible to association problems. The approach in this chapter tries to avoid this by using soft detection.

Next best view planning is applied to detect furniture parts in Blodow et al. [2011] and object on a table in Atanasov et al. [2013]. Maps consisting only of semantic entities (relational object maps) have been introduced in Limketkai et al. [2005] by modeling spaces with *relational Markov networks*. In Ekvall et al. [2007], receptive Field Co-occurrence Histograms are used to detect object instances over multiple viewpoints obtained from SLAM. Objects and supporting planes are estimated concurrently by Bao et al. [2010]. Plane normals are derived from object pose and the optimal solution is found by an exhaustive search carried out via parallel programming. Sengupta et al. [2013] derive semantic segmentation for a scene by classifying voxels, emphasizing 3D euclidean accuracy over

semantics.

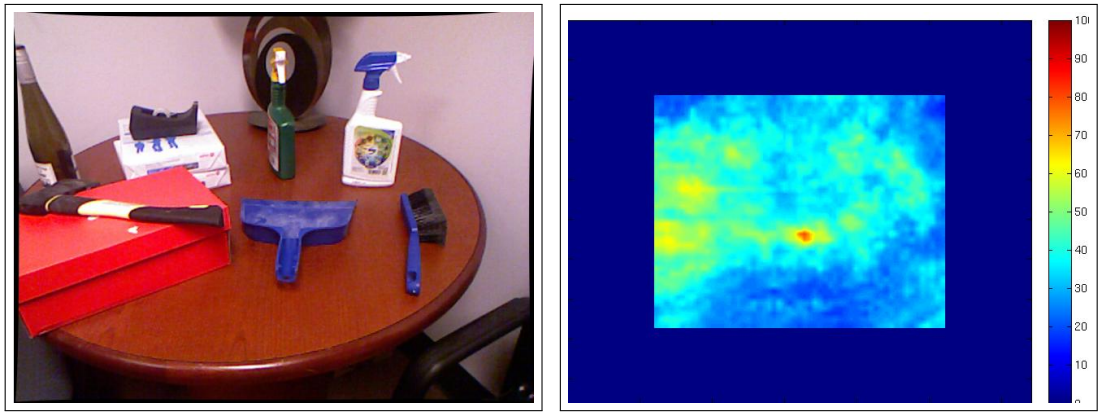
The most related approach is Bao et al. [2012] where objects are detected in the environment from pairs of views while simultaneously solving for camera pose and region (planar surfaces) information. In contrast with their approach, simple templates are used instead of point features and do not make any hard decisions in either 2D or 3D. Although their approach is more comprehensive including object class learning, structure from motion and region segmentation, it only incorporates information from two images while this chapter allows for much longer sequence.

5.3 Object Detection

Traditional object detectors yield one of the following representations when processing a query image: boolean flag [Lazebnik et al., 2006], bounding boxes [Felzenszwalb et al., 2010], and full object (or clutter) segmentation [Toshev et al., 2010]. These often include a score (or probability) measure of the detection as a whole, but always culminate in a hard decision as to the presence of an object (i.e., indicated by a bounding box).

Instead of relying on bounding boxes or image segmentation, this system operates on object heatmaps. These are maps indicating the likelihood of the object being present. They provide a value for each pixel (or block of pixels) and give a dense detection response for a given query image, as opposed to a sparse detection response. This likelihood is normalized in a global fashion to ensure that results for different templates and different features are comparable.

The highly optimized template matching algorithm from Hinterstoisser et al. [2010] is used. It is designed to be resilient to a small degree of object appearance variability. Each object is represented by an image template, $\mathcal{T}$, that captures its appearance in terms of a set of local histograms of gradients. The similarity score, s, between a template, $\mathcal{T}$, and the corresponding image region centered at location $\mathbf{x} = (x, y)$ in the search image, $\mathcal{I}$, is



(a) (b)

Figure 5.1: Examples of a score map generated by the DOT detector. (a) The input image, (b) DOT response map for class dust-pan.

computed by,

$$s(\mathbf{x}) = \sum_{\mathbf{r}\in\Omega}\left(\max_{\mathbf{x}'\in\mathcal{R}(\mathbf{x}+\mathbf{r})} S(\mathcal{T}(\mathbf{r}), \mathcal{I}(\mathbf{x}')\right), \tag{5.1}$$

where Ω denotes the set of template points, $\mathcal{R}(\mathbf{x}+\mathbf{r})$ is a local image neighbourhood centered on image location $\mathbf{x}+\mathbf{r}$ of image $\mathcal{I}$ which allows for slight image variations and $S(\mathcal{T}(\mathbf{r}), \mathcal{I}(\mathbf{x}'))$ measures the similarity between corresponding regions in the template and search images (see [Hinterstoisser et al., 2010] for details). Figure 5.1 contains an example image, and its associated similarity score for the object category dust-pan.

5.4 Computing the Map

Instead of defining the map of the environment as a collection of landmarks or objects, a voxel representation is used. By using a voxel representation, the system does not commit to a fixed number of detections at the onset, as in Bao et al. [2011a]. Nor does it have to explicitly model all possible configurations and data associations as in Atanasov et al. [2013]. Instead, data association is postponed until the map recovery step (section 5.4).

The voxels represent a fixed volume in 3D space defined by a box in space and the

individual voxels' dimensions. For simplicity the voxels are uniformly sized. Each voxel v contains aggregate object detector response values for all object classes $\mathcal{C}$. Then,

$$v = \left(s_1, \ldots, s_{|\mathcal{C}|}\right),$$

is the ordered list of detector scores for all classes $\mathcal{C}$ at a given voxel, and

$$\mathcal{V}_c = \{v(c) \mid \forall v \in \mathcal{V}\},$$

the set of scores from all voxels for a given class c. $\mathcal{V}$ is updated incrementally with each new incoming image. These incremental changes are noted with a frame id k, so $\mathcal{V}^k$ denotes the state of the voxel volume after k frames.

To determine which voxels in space are dependent on a given score map, the dense DOT result is back projected into the volume. This projection is implemented by projecting the score-map into the volume on a plane-by-plane basis. The volume is divided into x-y slices, creating a stack of planes parallel to the ground. Using xy-planes for this division not only simplifies the derivation, but is also intuitive.

The values from the response map are then interpolated for each voxel contained in the current plane. This volume represents the contribution of the detector for a given class in the current image to the 3D space. To integrate new information from additional frames, new volumes are added to the previous one, resulting in a cumulative sum of object detector responses. The final result is a Hough volume for each object within the space. By integrating the entire response volume directly, the system is able to accumulate suboptimal detector responses across multiple views in order to better identify and localize objects in the scene.

Only when the last frame is integrated into the volume is a threshold applied. The final 3D representation is constructed using non-maxima suppression to locate promising candidates. Given an approximate bounding volume of the object, search for and record local maxima. At each maximal location we remove all responses falling within the bounding volume centered around our current candidate. This process is repeated until the remaining scores within the volume fall below a threshold.

5.5 Experiments

A small office environment was captured to serve as the data set for this experiment. The data is composed of several sequences of images captured from a Kinect camera, and the recovered 6DOF camera pose. The pose was recovered using RGBD-SLAM [Engelhard et al., 2011], after which all depth information was discarded. Note that any odometry system which is able to recover 6DOF camera pose could be used to substitute for RGBD-SLAM.

The camera is hand-held and help mostly upright (minimal rotation around optical axis) but otherwise undergoes unconstrained motion. In these experiments, a spatial resolution of $1cm^3$ per voxel. The base of the volume is 2×2 meters and $70cm$ high.

5.5.1 Target Objects

The system is capable of using both textured and untextured objects. Use of the DOT object detector favours outlined and shape information.

To acquire the DOT training images a 3D model of the object was used. This model is rendered by placing the virtual camera at various points around a view sphere centered on the object. A total of 144 views are rendered covering 36 bearing angles θ and four elevation angles ϕ. Additionally, the model is rendered at two distances to account for some scale variation. Each rendered image is then fed to the DOT template trainer.

5.5.2 Camera Pose

Recall that camera pose is recovered using the RGBD-SLAM [Engelhard et al., 2011] package for ROS. In RGBD-SLAM, 2D features (in this case SURF [Bay et al., 2008]) are extracted and matched across image pairs. The corresponding depth value at each interest point is used to triangulate them. Once triangulated, the initial relative position and orientation between the two camera poses is recovered using RANSAC. This pose is refined using Iterative Closest Point (ICP). Finally a globally consistent pose is found

using hierachical pose graph optimizations (HOG-Man [Grisetti et al., 2010]).

Using the depth of features matched across frames the relative positions of the cameras are calculated using RANSAC. The first frame serves as a reference and as origin for world coordinate frame. To simplify correspondence between the world xy-plane and the camera the Kinect was placed on a flat surface to capture this first frame. The output is a list of timestamped camera poses which are used directly. After acquiring this list of poses the depth information is discarded.

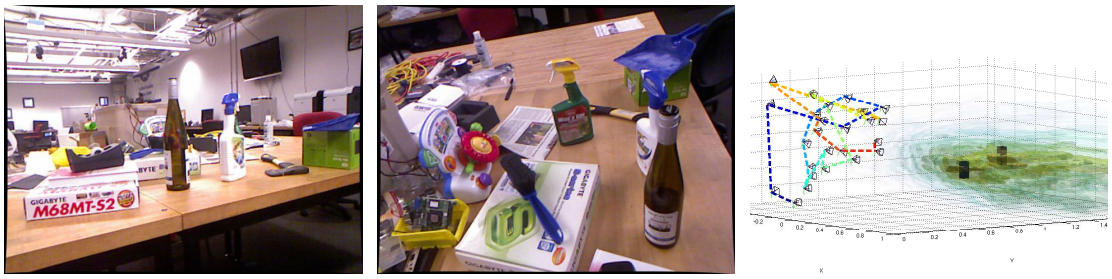
5.5.3 Results

Processing a sequence begins with processing each frame with all trained DOT templates. As each input image is processed by the DOT detector, it is back-projected into the volume of interest using the provided odometry. The resulting score-map is projected onto the voxel representation at each plane. Back-projected volumes from different frames are added together to yield a cumulative score.

Figures 5.2 and 5.3 show examples of a projected volume with detections for class spray-bottle and bottle respectively. In figure 5.2 the top row includes the first and sixth images of a sequence, while the bottom row is the final volumetric map for class spray-bottle, with the maximal detections represented by solid boxes. Figure 5.3 similarly includes the first and 11th images from the sequence, with the bottom row showing the volumetric map for the class bottle. The position and orientation of the camera frames are included for reference.

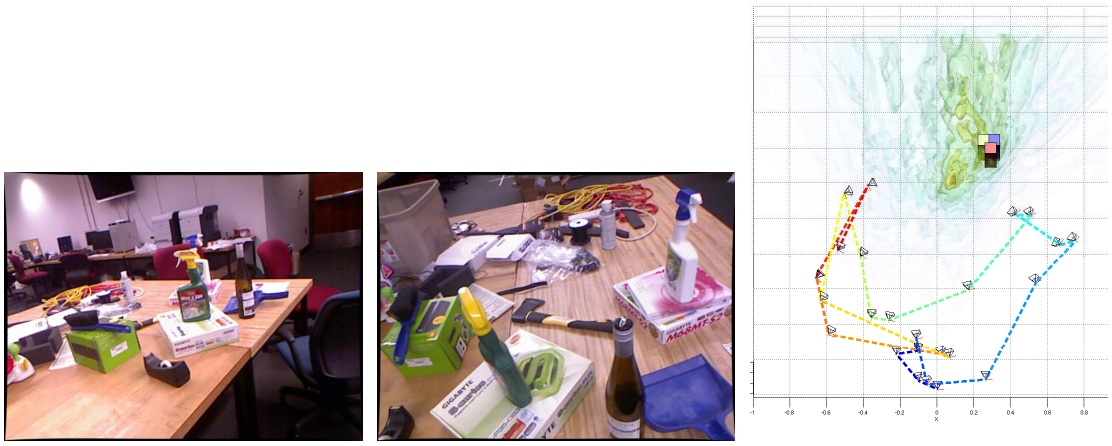
5.6 Summary

This chapter presented a system which uses a dense response map of object detections in images to localize objects. Although it employs a naive model, it is still able to recover objects in the scene. Additionally using a 3D voxel representation eliminates the need for a list of candidate hypotheses. By not committing to a fixed set of object candidates the



(a) (b) (c)

Figure 5.2: 3D Plot of spray-bottle detections super-imposed over reconstructed Hough volume. The input camera frames are shown connected through time from start (blue) to finish (red). Show are the first frame (a), an intermediate frame showing two spray-bottles originally hidden (b) and the resulting volume with detections (c).



(a) (b) (c)

Figure 5.3: 3D Plot of bottle detections super-imposed over reconstructed Hough volume. The input camera frames are shown connected through time from start (blue) to finish (red). Show are the first frame (a), an intermediate frame showing a closeup of the the bottle (b) and the resulting volume with detections (c).

system is free to integrate all the data across multiple frames. This allows it to recover objects that may otherwise undetected in each individual image.

The approach relies on odometry input, the next logical step is eliminating this prerequisite by extending the system to recover the odometry. Ideally, the camera pose should be recovered using objects so as to take advantage of the semantic information already recovered during the mapping step. Object based localization such as presented in chapter 4 could be employed iteratively with this work to improve both localization and mapping. This could then be further enhanced using other traditional odometry algorithms for robustness.

Chapter 6

Conclusions

This dissertation presented three advancements in vision based semantic mapping and localization. The emphasis was placed on devising approaches that distinguish themselves by not falling into the categories listed in chapter 1:

- Not pure vision.
- 3D metric approaches.
- Scene *classification* instead of true *localization*.
- Limited, or no semantics.

Pure vision approaches have an advantage over systems that incorporate other sensors since they can work in most environments. While GPS cannot function indoors, and infra-red sensors cannot function outdoors, a regular camera functions both indoors and outdoors. This combined with the explosion in mobile digital cameras, make pure vision approaches easy to deploy and functional in any situation.

3D metric solutions require and provide high-precision which, depending on the application, may entail unnecessary complexity. Approximate solutions like those presented here often suffice, and are not only cheaper but can also be more accessible to human interaction. However, sacrificing spatial accuracy does not mean ignoring spatial information

altogether. The proposed approaches do incorporate spatial information and thus provide usable localization and mapping. Finally, the inclusion of semantic reasoning, input, and output promotes human interaction with the proposed systems.

Chapter 3 described a novel method for creating topological maps from video streams. The primary contribution of the work was a novel image similarity score, which when coupled with an MRF representation allowed for accurate recovery of loop-closures. The resulting loop-closures were then fed into algorithm 3.1 to produce the final topological map. The approach combined purely visual information to create a non-metric map of an environment while maintaining connectivity between adjacent location At the same time is also clustered images into individual semantically meaningful location.

The localization approach of chapter 4 used an existing map with labeled objects to approximately localize the robot. Images from the robot are processed by an object detector to yield soft object heatmaps. These soft detections are then directly used as input into a particle filter localization scheme (section 4.5) or as part of a probabilistic observational model in section 4.8. As the robot moves, more information is integrated into its location hypothesis yielding a final localization. The results showed that by using a visual sensor it was possible to integrate semantic information in the form of object detections to yield the location of the camera. Although the resulting location is absolute with respect to the input map, it is not metric in that it relies solely on the relative bearing of objects as measured from the local camera frame.

Finally the novel 3D object recognition and localization framework presented in chapter 5. Images from a moving camera are processed with an object detector to generate object heatmaps. Heatmaps are back-projected into 3D space to generate volumetric heatmaps. Qualitative results showed that the system is able to identify objects in cluttered environments. The maps generated by this system can also be employed to boot-strap the localization approach from chapter 4.

Together the proposed approaches each address a separate yet complimentary aspect of semantic understanding of environments. Topological maps can be used to navigate large

scale environments. They overlay a discrete network graph on the mapped space resulting in a discretization of the environment allowing for approximate localization. Finer grained positioning can then be recovered with object-based localization. Using the surrounding objects to approximate the robot's position in medium to small scale environments. This level of localization requires a semantic map of the immediate surroundings. Such a map can be constructed using the proposed 3D object recognition system. Indeed, all three approaches together can be combined into a hybrid localization framework in the vein of Bosse et al. [2003]; Tomatis et al. [2003] and Drouilly et al. [2014].

Another key aspect of this dissertation is the primacy of objects as mid-level features. Recall that chapter 3 relies on low-level features for computing image similarity. On the other hand, chapter 4 uses object detections instead. While it's possible to perform localization with low-level features, mid-level features bring with them several advantages. Foremost, using object detections as features greatly simplifies the underlying representation. An annotated semantic map can be easily generated from a hand-drawn sketch, whereas using low-level features for localization requires costly SLAM algorithms. Semantic maps are also simpler for knowledge transfer; instead of thousands of highly localized features, a simple handful of roughly placed semantic labels suffice. This allows separation of mapping from localization.

Traditional localization with low-level features essentially maps an environment and then aligns the recovered map to a reference map. Using objects as features permits localization *without* mapping. Maps can be generated apriori using sensors or data not available during localization. By abstracting the contents of the map to semantic annotations, localization can be performed with different sensors. Maps produced with expensive platforms can be used by commodity cameras, semantic maps produced today are still relevant tomorrow when better object detectors become available. Finally, separating semantic mapping from semantic localization enables human interaction. Semantic maps can be generated by hand and still be used for automatic localization. On the other hand, automatically generated semantic maps can be used for manual localization. These use

cases are not possible with just low-level features.

6.1 Future Directions

The continued development of semantic solutions to problems in robotics is producing human meaningful representations of data, and incorporating human logic and reasoning into automated systems. Further research in mapping, localization, and detection needs to incorporate more human concepts and constructs. In addition to increasing semantics these will also need to peel away constraints and assumptions in order to narrow the "semantic gap".

In mapping, significant results exist both for indoor and outdoor settings, but few approaches exist for creating maps combining both, a distinction that is essential for everyday human activity. A similar lack exists for localization approaches, with indoor systems focusing on architecture and objects while outdoor systems rely on GPS, odometry and appearance. Although some work exists for automatically incorporating text in indoor settings [Case et al., 2011] and manually incorporating text for outdoor localization (e.g. using ReCaptcha on StreetView images), an automated approach does not exist for outdoor localization using signage. For robots to be of use in more scenarios, it is necessary to continue expanding semantics to more complex levels. The proliferation in recent years of convolutional neural network (CNN) based approaches have resulted in significant performance gains in object detection and localization. As object detectors these can prove a valuable source for soft-object based approaches. These can also be tailored and trained to soft-object based solutions directly, providing accurate and dense object heatmaps. In addition to improved detector performance, semantics in the form of object functionality is also crucial for future applications. Grabner et al. [2011] are already detecting and learning functionality using simulation with a virtual actor (as opposed to explicit training videos). However, in this approach and others, the robot is still a passive participant, exploring and mapping but not interacting. Thus we should seek to expand semantics to

incorporate functionality for the purpose of actual interaction as in Beetz et al. [2011].

For computers and robots to function as autonomous agents in a human world, human semantics are essential. By both easing communication and incorporating more complex human type reasoning, the class of environments and tasks these machines can cope with grows. These advancements will bring with them autonomous cars, robotic cooks, and computerized assistants which will be better able to interact with human users without requiring us to drastically modify our environments to permit their functionality.

Bibliography

R. Anati and K. Daniilidis. Constructing topological maps using Markov random fields and loop-closure detection. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Neural Information Processing Systems (NIPS)*, pages 37–45. Curran Associates, Inc., 2009. 7

R. Anati, D. Scaramuzza, K. Derpanis, and K. Daniilidis. Robot localization using soft object detection. In *IEEE International Conference on Robotics and Automation (ICRA)*, May 2012. doi:10.1109/ICRA.2012.6225216. ©International Conference on Robotics and Automation IEEE. Reprinted, with permission from the authors. 7

A. Angeli, D. Filliat, S. Doncieux, and J. A. Meyer. Fast and incremental method for loop-closure detection using bags of visual words. *IEEE Transactions on Robotics (T-RO)*, 24(5):1027–1037, Sept. 2008. doi:10.1109/TRO.2008.2004514. 9, 26

D. Anguelov, R. Biswas, D. Koller, B. Limketkai, and S. Thrun. Learning hierarchical object maps of non-stationary environments with mobile robots. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 10–17. Morgan Kaufmann Publishers Inc., 2002. 4, 17, 122, 123

I. Apostolopoulos, N. Fallah, E. Folmer, and K. E. Bekris. Integrated online localization and navigation for people with visual impairments using smart phones. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2012. doi:10.1109/ICRA.2012.6225093. 1, 4, 13, 15, 16, 17

G. Arbeiter, M. Haegele, and A. Verl. Field of view dependent registration of point clouds and incremental extraction of table-tops using time-of-flight cameras. In *IEEE International Conference on Robotics and Automation (ICRA), Workshop on Semantic Perception, Mapping and Exploration (SPME)*, 2011. 18

N. Atanasov, B. Sankaran, J. Le Ny, T. Koletschka, G. Pappas, and K. Daniilidis. Hypothesis testing framework for active object detection. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2013. doi:10.1109/ICRA.2013.6631173. 122, 123, 125

N. Atanasov, M. Zhu, K. Daniilidis, and G. Pappas. Semantic localization via the matrix permanent. In *Proceedings of Robotics: Science and Systems (RSS)*, 2014. 14, 16, 51, 80

N. Atanasov, M. Zhu, K. Daniilidis, and G. J. Pappas. Localization from semantic observations via the matrix permanent. *The International Journal of Robotics Research (IJRR)*, 35(1–3):73–99, 2016. doi:10.1177/0278364915596589. URL http://ijr.sagepub.com/content/35/1-3/73.abstract. 14, 16, 51

M. Bader and M. Vincze. *RoboCup 2013: Robot World Cup XVII*, chapter Spontaneous Reorientation for Self-localization, pages 456–467. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014. ISBN 978-3-662-44468-9. doi:10.1007/978-3-662-44468-9$_4$0. 13

M. Bader, J. Prankl, and M. Vincze. Visual room-awareness for humanoid robot self-localization. abs/1304.5878, 2013. URL http://arxiv.org/abs/1304.5878. 13

S. Y. Bao and S. Savarese. Semantic structure from motion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. doi:10.1109/CVPR.2011.5995462. 18

S. Y. Bao, M. Bagra, and S. Savarese. Semantic structure from motion with object and

point interactions. In *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, 2011a. doi:10.1109/ICCVW.2011.6130358. 125

S. Y. Bao, M. Sun, and S. Savarese. Toward coherent object detection and scene layout understanding. *Image and Vision Computing*, 29(9):569–579, Aug. 2011b. ISSN 0262-8856. doi:10.1016/j.imavis.2011.08.001. 18

S. Y. Bao, M. Bagra, Y.-W. Chao, and S. Savarese. Semantic structure from motion with points, regions, and objects. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 4, 17, 18, 20, 21, 122, 124

S. Y.-Z. Bao, M. Sun, and S. Savarese. Toward coherent object detection and scene layout understanding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. doi:10.1109/CVPR.2010.5540229. 18, 19, 21, 122, 123

H. Bay, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 404–417, 2006. 14

H. Bay, T. Tuytelaars, and L. V. Gool. SURF: Speeded up robust features. *Computer Vision and Image Understanding (CVIU)*, 110(3):346–359, 2008. 127

P. Beeson, N. K. Jong, and B. Kuipers. Towards autonomous topological place detection using the extended voronoi graph. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4373–4379, 2005. doi:10.1109/ROBOT.2005.1570793. 9, 10, 12

M. Beetz, U. Klank, I. Kresse, A. Maldonado, L. Mosenlechner, D. Pangercic, T. Ruhr, and M. Tenorth. Robotic roommates making pancakes. In *Humanoid Robots*, 2011. 135

A. Bhattacharyya. On a measure of divergence between two statistical populations defined by their probability distribution. *Bulletin of the Calcutta Mathematical Society*, 35:99–110, 1943. 56

C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2006. 32, 64

N. Blodow, L. C. Goron, Z.-C. Marton, D. Pangercic, T. Ruehr, M. Tenorth, and M. Beetz. Inferring generalized pick-and-place tasks from pointing gestures. In *IEEE International Conference on Robotics and Automation (ICRA), Workshop on Semantic Perception, Mapping and Exploration (SPME)*, 2011. 122, 123

O. Booij, Z. Zivkovic, and B. Kröse. Pruning the image set for appearance based robot localization. In *Conference of the Advanced School for Computing and Imaging (ASCI)*, pages 57–64, June 2005. 27, 32

O. Booij, B. Terwijn, Z. Zivkovic, and B. Krose. Navigation using an appearance based topological map. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3927–3932, 2007. doi:10.1109/ROBOT.2007.364081. 13, 27

O. Booij, Z. Zivkovic, and B. Kröse. Efficient data association for view based SLAM using connected dominating sets. *Robotics and Autonomous Systems (RAS)*, 57(12):1225–1234, 2009. ISSN 0921-8890. doi:10.1016/j.robot.2009.06.006. URL http://www.sciencedirect.com/science/article/pii/S0921889009000888. 9

M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten, and S. Teller. An atlas framework for scalable mapping. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 1899–1906, 2003. doi:10.1109/ROBOT.2003.1241872. 3, 9, 10, 11, 12, 13, 26, 133

T. Botterill, S. Mills, and R. Green. Correcting scale drift by object recognition in single-camera slam. *IEEE Transactions on Cybernetics*, 43(6):1767–1780, Dec. 2013. ISSN 2168-2267. doi:10.1109/TSMCB.2012.2230164. 13

W. Burgard and M. Hebert. World modelling. In *Handbook of Robotics*. Springer, 2008. 122

C. Case, B. Suresh, A. Coates, and A. Y. Ng. Autonomous sign reading for semantic mapping. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011. 18, 134

R. O. Castle and D. W. Murray. Object recognition and localization while tracking and mapping. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, 2009. 4, 17, 21

R. O. Castle, D. J. Gawley, G. Klein, and D. W. Murray. Towards simultaneous recognition, localization and mapping for hand-held and wearable cameras. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4102–4107, 2007. 17

R. O. Castle, G. Klein, and D. W. Murray. Video-rate localization in multiple maps for wearable augmented reality. In *IEEE International Symposium on Wearable Computers (ISWC)*, pages 15–22, 2008. 19

R. O. Castle, G. Klein, and D. W. Murray. Combining monoslam with object recognition for scene augmentation using a wearable camera. *Image and Vision Computing*, 28(11): 1548–1556, Nov. 2010. doi:10.1016/j.imavis.2010.03.009. 18, 19

H. Choset and K. Nagatani. Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization. *IEEE Transactions on Robotics and Automation (T-RA)*, 17(2):125–137, Apr. 2001. 9

H. Choset, S. Walker, K. Eiamsa-Ard, and J. Burdick. Sensor-based exploration: Incremental construction of the hierarchical generalized voronoi graph. *The International Journal of Robotics Research (IJRR)*, 19(2):96–125, 2000. doi:10.1177/02783640022066789. 9, 10, 12

V. Chvatal. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4(3):233–235, 1979. URL http://www.jstor.org/stable/3689577. 32

J. Civera, D. Gálvez-López, L. Riazuelo, J. Tardós, and J. Montiel. Towards semantic SLAM using a monocular camera. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2011. doi:10.1109/IROS.2011.6094648. 18, 52, 122, 123

J. Cleveland, D. Thakur, P. Dames, C. Phillips, T. Kientz, K. Daniilidis, J. Bergstrom, and V. Kumar. An automated system for semantic object labeling with soft object recognition and dynamic programming segmentation. In *IEEE International Conference on Automation Science and Engineering (CASE)*, pages 683–690, Aug. 2015. doi:10.1109/CoASE.2015.7294159. 19, 119, 122

M. Cummins and P. Newman. Fab-MAP: Probabilistic localization and mapping in the space of appearance. *The International Journal of Robotics Research (IJRR)*, 27(6): 647–665, June 2008a. doi:10.1177/0278364908090961. 2, 11, 12, 25, 26, 33, 45

M. Cummins and P. Newman. Accelerated appearance-only SLAM. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1828–1833, 2008b. doi:10.1109/ROBOT.2008.4543473. 2, 11, 12, 34

M. Cummins and P. Newman. Highly scalable appearance-only SLAM - fab-MAP 2.0. In *Proceedings of Robotics: Science and Systems (RSS)*, Seattle, USA, June 2009. 11, 12, 26, 28

M. Cummins and P. Newman. Accelerating fab-MAP with concentration inequalities. *IEEE Transactions on Robotics (T-RO)*, 26(6):1042–1050, Dec. 2010a. doi:10.1109/TRO.2010.2080390. 11, 12, 26, 46

M. Cummins and P. Newman. Appearance-only SLAM at large scale with FAB-MAP 2.0. *The International Journal of Robotics Research (IJRR)*, 30(9):1100–1123, Nov. 2010b. 10, 11, 12, 26

F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte Carlo localization for mobile robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, 1999. doi:10.1109/ROBOT.1999.772544. 47, 58

B. Douillard, D. Fox, Ramos, and H. Durrant-Whyte. Classification and semantic mapping of urban environments. *The International Journal of Robotics Research (IJRR)*, 30(1): 5–32, 2010. doi:10.1177/0278364910373409. 18

R. Drouilly, P. Rives, and B. Morisset. Fast hybrid relocation in large scale metric-topologic-semantic map. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 1839–1845, Sept. 2014. doi:10.1109/IROS.2014.6942804. 13, 16, 133

S. Ekvall, P. Jensfelt, and D. Kragic. Integrating active mobile robot object recognition and SLAM in natural environments. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 5792–5797, Oct. 2006. doi:10.1109/IROS.2006.282389. 13, 17

S. Ekvall, D. Kragic, and P. Jensfelt. Object detection and mapping for service robot tasks. *Robotica*, 25(2):175–187, 2007. 122, 123

N. Engelhard, F. Endres, J. Hess, J. Sturm, and W. Burgard. Real-time 3D visual SLAM with a hand-held rgb-d camera. In *Proceedings of the RGB-D Workshop on 3D Perception in Robotics*, Vasteras, Sweden, Apr. 2011. 121, 127

O. Erkent and I. Bozma. Place representation in topological maps based on bubble space. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2012. doi:10.1109/ICRA.2012.6225367. 10

P. Espinace, T. Kollar, A. Soto, and N. Roy. Indoor scene recognition through object detection. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1406–1413, 2010. doi:10.1109/ROBOT.2010.5509682. 13, 15, 16, 52

M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision (IJCV)*, 88(2):303–338, June 2010. 36

P. F. Felzenszwalb, R. B. Girshick, and D. McAllester. Discriminatively trained deformable part models, release 4. URL http://cs.brown.edu/~pff/latent-release4/. 81

P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, 32(9):1627–1645, 2010. doi:10.1109/TPAMI.2009.167. 53, 64, 71, 124

F. Fraundorfer, C. Engels, and D. Nister. Topological mapping, localization and navigation using image collections. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2007. doi:10.1109/IROS.2007.4399123. 2, 3, 9, 12, 13

F. Fraundorfer, C. Wu, J. M. Frahm, and M. Pollefeys. Visual word based location recognition in 3D models using distance augmented weighting. In *Proceedings of IEEE International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, 2008. 27

W. Freeman and E. Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, 13(9):891–906, Sept. 1991. doi:10.1109/34.93808. 55

C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J. A. Fernández-Madrigal, and J. González. Multi-hierarchical semantic maps for mobile robotics. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 2278–2283. IEEE, 2005. ISBN 0780389123. doi:10.1109/IROS.2005.1545511. 1, 52

X. Gao, X. Hou, J. Tang, and H. Cheng. Complete solution classification for the perspective-three-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, 25(8):930–943, 2003. doi:10.1109/TPAMI.2003.1217599. 58

E. Garcia-Fidalgo and A. Ortiz. Vision-based topological mapping and localization by means of local invariant features and map refinement. *Robotica*, 33:1446–1470,

Aug. 2015a. ISSN 1469-8668. doi:10.1017/S0263574714000782. URL http://journals.cambridge.org/article_S0263574714000782. 10

E. Garcia-Fidalgo and A. Ortiz. Vision-based topological mapping and localization methods: A survey. *Robotics and Autonomous Systems (RAS)*, 64:1 – 20, 2015b. ISSN 0921-8890. doi:10.1016/j.robot.2014.11.009. URL http://www.sciencedirect.com/science/article/pii/S0921889014002619. 8

T. Goedemé, M. Nuttin, T. Tuytelaars, and L. Van Gool. Omnidirectional vision based topological navigation. *International Journal of Computer Vision (IJCV)*, 74(3):219–236, Sept. 2007. doi:10.1007/s11263-006-0025-9. 9, 27

H. Grabner, J. Gall, and L. V. Gool. What makes a chair a chair? In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. doi:10.1109/CVPR.2011.5995327. 134

G. Grisetti, R. Kuemmerle, C. Stachniss, U. Frese, and C. Hertzberg. Hierarchical optimization on manifolds for online 2d and 3d mapping. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2010. URL https://openslam.org/hog-man.html. 128

M. Gunther, T. Wiemann, S. Albrecht, and J. Hertzberg. Building semantic object maps from sparse and noisy 3d data. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 2228–2233, Nov. 2013. doi:10.1109/IROS.2013.6696668. 122, 123

P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments. In *International Symposium on Experimental Robotics(ISER)*, 2010. 121

S. Hinterstoisser, V. Lepetit, S. Ilic, P. Fua, and N. Navab. Dominant orientation templates for real-time detection of texture-less objects. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2257–2264, 2010. doi:10.1109/CVPR.2010.5539908. 121, 124, 125

S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, 2012. 123

Ho, Kin, Newman, and Paul. Detecting loop closure with scene sequences. *International Journal of Computer Vision (IJCV)*, 74(3):261–286, Sept. 2007. ISSN 0920-5691. doi:10.1007/s11263-006-0020-1. 9

R. Ivanov, N. Atanasov, M. Pajic, I. Lee, and G. Pappas. Robust localization using context-aware filtering. In *Workshop on Multi-View Geometry in Robotics (MVIGRO 2015) in conjunction with The 2015 Robotics: Science and Systems Conference*, 2015. 13

T. Kanji. Cross-season place recognition using nbnn scene descriptor. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 729–735, Sept. 2015a. doi:10.1109/IROS.2015.7353453. 14

T. Kanji. Unsupervised part-based scene modeling for visual robot localization. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6359–6365, May 2015b. doi:10.1109/ICRA.2015.7140092. 14

N. Kejriwal, S. Kumar, and T. Shibata. High performance loop closure detection using bag of word pairs. *Robotics and Autonomous Systems*, 77:55–65, 2016. ISSN 0921-8890. doi:10.1016/j.robot.2015.12.003. URL http://www.sciencedirect.com/science/article/pii/S0921889015300889. 10

G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In

International Symposium on Mixed and Augmented Reality (ISMAR), Nara, Japan, Nov. 2007. 9, 12, 13

G. Klein and D. Murray. Parallel tracking and mapping on a camera phone. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, Orlando, Oct. 2009. 9, 13

L. Kneip, D. Scaramuzza, and R. Siegwart. A novel parameterization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. doi:10.1109/CVPR.2011.5995464. 58

D. W. Ko, C. Yi, and I. H. Suh. Semantic mapping and navigation: A bayesian approach. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 2630–2636, Nov. 2013. doi:10.1109/IROS.2013.6696727. 18

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009. 32

K. Konolige and M. Agrawal. Frameslam: From bundle adjustment to real-time visual mapping. *IEEE Transactions on Robotics (T-RO)*, 24(5):1066–1077, 2008. doi:10.1109/TRO.2008.2004832. 9

H. Korrapati and Y. Mezouar. Multi-resolution map building and loop closure with omnidirectional images. *Autonomous Robots*, pages 1–21, 2016. ISSN 1573-7527. doi:10.1007/s10514-016-9560-6. 10

I. Kostavelis and A. Gasteratos. Learning spatially semantic representations for cognitive robot navigation. *Robotics and Autonomous Systems (RAS)*, 61(12):1460–1475, 2013. ISSN 0921-8890. doi:10.1016/j.robot.2013.07.008. URL `http://www.sciencedirect.com/science/article/pii/S0921889013001346`. 17, 18

B. J. Kuipers. The spatial semantic hierarchy. *Artificial Intelligence*, 119(1–2):191–233, May 2000. doi:10.1016/S0004-3702(00)00017-5. 51, 122

A. Kumar, J.-P. Tardif, R. Anati, and K. Daniilidis. Experiments on visual loop closing using vocabulary trees. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops)*, pages 1–8. IEEE, June 2008. ISBN 978-1-4244-2339-2. doi:10.1109/CVPRW.2008.4563140. 3

K. Lai, L. Bo, X. Ren, and D. Fox. A scalable tree-based approach for joint object and pose recognition. In *Conference on Artificial Intelligence (AAAI)*, 2011a. 18, 20, 21

K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view RGB-D object dataset. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011b. doi:10.1109/ICRA.2011.5980382. URL `http://www.cs.washington.edu/rgbd-dataset/`. 81, 83, 84

K. Lai, L. Bo, X. Ren, and D. Fox. Detection-based object labeling in 3D scenes. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2012. doi:10.1109/ICRA.2012.6225316. 4, 17, 20, 21

K. Lai, L. Bo, and D. Fox. Unsupervised feature learning for 3d scene labeling. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3050–3057, 2014. doi:10.1109/ICRA.2014.6907298. 19, 81, 83, 84, 86, 87, 119

Y. Latif, G. Huang, J. Leonard, and J. Neira. An online sparsity-cognizant loop-closure algorithm for visual navigation. In *Proceedings of Robotics: Science and Systems (RSS)*, Berkeley, USA, July 2014. 10, 11, 12, 27

S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2169–2178, 2006. doi:10.1109/CVPR.2006.68. 53, 124

H. Li, E. Kim, X. Huang, and L. He. Object matching with a locally affine-invariant constraint. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1641–1648, 2010a. doi:10.1109/CVPR.2010.5539776. 52

L.-J. Li, H. Su, E. Xing, and L. Fei-Fei. Object bank: A high-level image representation for scene classification and semantic feature sparsification. In *Neural Information Processing Systems (NIPS)*, 2010b. 1, 17, 18, 21, 22

B. Limketkai, L. Liao, and D. Fox. Relational object maps for mobile robots. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, volume 19, page 1471, 2005. 9, 53, 122, 123

Liu, Ziyuan, and G. von Wichert. Extracting semantic indoor maps from occupancy grids. *Robotics and Autonomous Systems (RAS)*, 62(5):663–674, 2014. ISSN 0921-8890. doi:10.1016/j.robot.2012.10.004. URL http://www.sciencedirect.com/science/article/pii/S092188901200187X. Special Issue Semantic Perception, Mapping and Exploration. 10

J. J. Liu, C. Phillips, and K. Daniilidis. Video-based localization without 3D mapping for the visually impaired. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops)*, 2010. doi:10.1109/CVPRW.2010.5543581. 13, 14, 16, 46

M. Liu, D. Scaramuzza, C. Pradalier, R. Siegwart, and Q. Chen. Scene recognition with omnidirectional vision for topological map using lightweight adaptive descriptors. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 116–121, Oct. 2009. doi:10.1109/IROS.2009.5354131. 9

Q. Liu, R. Li, H. Hu, and D. Gu. Extracting semantic information from visual data: A survey. *Robotics*, 5(1):8, 2016. ISSN 2218-6581. doi:10.3390/robotics5010008. URL http://www.mdpi.com/2218-6581/5/1/8. 17

D. G. Lopez, K. Sjo, C. Paul, and P. Jensfelt. Hybrid laser and vision based object search and localization. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2636–2643, 2008. doi:10.1109/ROBOT.2008.4543610. 4, 18, 19, 21

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004. doi:10.1023/B:VISI.0000029664.99615.94. 19, 28

W. L. D. Lui and R. Jarvis. A pure vision-based topological SLAM system. *The International Journal of Robotics Research (IJRR)*, 31(4):403–428, Apr. 2012. doi:10.1177/0278364911435160. 10

McManus, Colin, Upcroft, Ben, Newman, and Paul. Learning place-dependant features for long-term vision-based localisation. *Autonomous Robots*, 39(3):363–387, 2015. ISSN 1573-7527. doi:10.1007/s10514-015-9463-y. 14

D. Meger, P. Forssen, K. Lai, S. Helmer, S. McCann, T. Southey, M. Baumann, J. Little, and D. Lowe. Curious george: An attentive semantic robot. *Robotics and Autonomous Systems (RAS)*, 56(6):503–511, 2008. 18

J. Modayil, P. Beeson, and B. Kuipers. Using the topological skeleton for scalable global metrical map-building. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, volume 2, 2004. 9

A. Murillo and J. Kosecka. Experiments in place recognition using gist panoramas. In *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 2196–2203, Oct. 2009. doi:10.1109/ICCVW.2009.5457552. 1, 3, 9, 14, 16

A. C. Murillo, J. Kosecka, J. J. Guerrero, and C. Sagues. Visual door detection integrating appearance and shape cues. *Robotics and Autonomous Systems (RAS)*, 56(6):512–521, 2008. ISSN 0921-8890. 52

A. C. Murillo, P. Campos, J. Kosecka, and J. J. Guerrero. Gist vocabularies in omnidirectional images for appearance based mapping and localization. In *The Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras (OMNIVIS)*, 2010. 13

A. C. Murillo, G. Singh, J. Koseck, and J. J. Guerrero. Localization in urban environments using a panoramic gist descriptor. 29(1):146–160, Feb. 2013. ISSN 1552-3098. doi:10.1109/TRO.2012.2220211. 13

D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2161–2168, 2006. doi:10.1109/CVPR.2006.264. 27, 46

A. Nüchter and J. Hertzberg. Towards semantic maps for mobile robots. *Robotics and Autonomous Systems (RAS)*, 56(11):915–926, 2008. 122, 123

J. Oberlander, K. Uhl, J. Zollner, and R. Dillmann. A region-based slam algorithm capturing metric, topological, and semantic properties. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1886–1891. IEEE, 2008. doi:10.1109/ROBOT.2008.4543482. 9, 53, 122, 123

A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision (IJCV)*, 42(3):145–175, 2001. doi:10.1023/A:1011139631724. 14, 46

I. Posner, D. Schroeter, and P. M. Newman. *Using Scene Similarity for Place Labelling*, volume 39, pages 85–98. Springer, Berlin / Heidelberg, 2008. 52

A. Pronobis. *Semantic Mapping with Mobile Robots*. PhD thesis, KTH Royal Institute of Technology, 2011. 2

A. Pronobis and P. Jensfelt. Large-scale semantic mapping and reasoning with heterogeneous modalities. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2012. doi:10.1109/ICRA.2012.6224637. 1, 3, 18

A. Pronobis, B. Caputo, P. Jensfelt, and H. I. Christensen. A realistic benchmark for visual indoor place recognition. *Robotics and Autonomous Systems (RAS)*, 58(1):81–96, Jan. 2010a. doi:10.1016/j.robot.2009.07.025. 3, 4, 13, 14, 16, 118, 122

A. Pronobis, J. Luo, and B. Caputo. The more you learn, the less you store: Memory-controlled incremental SVM for visual place recognition. *Image and Vision Computing*, 28(7):1080–1097, July 2010b. doi:10.1016/j.imavis.2010.01.015. 13

A. Pronobis, O. Mozos, B. Caputo, and P. Jensfelt. Multimodal semantic place classification. *The International Journal of Robotics Research (IJRR)*, 29(2–3):298–320, Feb. 2010c. doi:10.1177/0278364909356483. 3, 13, 14, 16, 18, 118

A. Ranganathan and F. Dellaert. Semantic modeling of places using objects. In *Proceedings of Robotics: Science and Systems (RSS)*, 2007. 51

A. Ranganathan, E. Menegatti, and F. Dellaert. Bayesian inference in the space of topological maps. *IEEE Transactions on Robotics (T-RO)*, 22(1):92–107, 2006. doi:10.1109/TRO.2005.861457. 9, 26

F. Ribeiro, S. Brando, J. P. Costeira, and M. Veloso. Global localization by soft object recognition from 3d partial views. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 3709–3714, Sept. 2015. doi:10.1109/IROS.2015.7353896. 14

A. Rituerto, A. C. Murillo, and J. J. Guerrero. Semantic labeling for indoor topological mapping using a wearable catadioptric system. *Robotics and Autonomous Systems (RAS)*, 2012. doi:10.1016/j.robot.2012.10.002. 19, 52

A. Rituerto, A. C. Murillo, and J. J. Guerrero. Semantic labeling for indoor topological mapping using a wearable catadioptric system. *Robotics and Autonomous Systems (RAS)*, 62(5):685–695, 2014. ISSN 0921-8890. doi:10.1016/j.robot.2012.10.002. URL http://www.sciencedirect.com/science/article/pii/S0921889012001856. Special Issue Semantic Perception, Mapping and Exploration. 10, 17, 19

A. Rottmann, Óscar Martínez Mozos, C. Stachniss, and W. Burgard. Semantic place classification of indoor environments with mobile robots using boosting. In *Conference on Artificial Intelligence (AAAI)*, pages 1306–1311. AAAI Press, 2005. ISBN 1-57735-236-x. URL http://dl.acm.org/citation.cfm?id=1619499.1619543. 3, 4, 13, 17, 118

O. Russakovsky and A. Y. Ng. A steiner tree approach to efficient object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2010. doi:10.1109/CVPR.2010.5540097. 1

R. Rusu and S. Cousins. 3d is here: Point cloud library (pcl). In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–4. IEEE, 2011. 122

R. Rusu, Z. Marton, N. Blodow, M. Dolha, and M. Beetz. Towards 3D point cloud based object maps for household environments. *Robotics and Autonomous Systems (RAS)*, 56 (11):927–941, 2008. 123

R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison. Slam++: Simultaneous localisation and mapping at the level of objects. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013. 13, 16, 18, 51, 52, 122, 123

T. Sattler, B. Leibe, and L. Kobbelt. Improving image-based localization by active correspondence search. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012. 13

D. Scaramuzza, N. Criblez, A. Martinelli, and R. Siegwart. Robust feature extraction and matching for omnidirectional images. In *International Conference on Field and Service Robotics (FSR)*, volume 42 of *Springer Tracts in Advanced Robotics*, Chamonix, France, Mar. 2008. Springer Press. 13, 15, 16

D. Schroter, M. Beetz, and J. Gutmann. Rg mapping: Learning compact and structured 2d line maps of indoor environments. In *Proceedings of the IEEE International Workshop on Robot and Human Interactive Communication (ROMAN)*, pages 282–287. IEEE, 2002. doi:10.1109/ROMAN.2002.1045636. 52

S. Sengupta, E. Greveson, A. Shahrokni, and P. H. S. Torr. Urban 3d semantic modelling using stereo vision. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 580–585, May 2013. doi:10.1109/ICRA.2013.6630632. 19, 122, 123

G. Singh and J. Kosecka. Acquiring semantics induced topology in urban environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2012. doi:10.1109/ICRA.2012.6225282. 1, 2, 10, 11, 12, 27

G. Singh and J. Košecká. Visual loop closing using gist descriptors in manhattan world. In *in Omnidirectional Robot Vision workshop, held with IEEE ICRA*, 2010. 10

B. Steder, G. Grisetti, and W. Burgard. Robust place recognition for 3D range data based on point features. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1400–1405, May 2010. doi:10.1109/ROBOT.2010.5509401. 13

S. Stoeter, F. Le Mauff, and N. Papanikolopoulos. Real-time door detection in cluttered environments. In *IEEE International Symposium on Intelligent Control (ISIC)*, pages 187–192. IEEE, 2000. doi:10.1109/ISIC.2000.882921. 52

A. Tapus and R. Siegwart. Incremental robot mapping with fingerprints of places. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 2429–2434, Aug. 2005. doi:10.1109/IROS.2005.1544977. 9

J.-P. Tardif, Y. Pavlidis, and K. Daniilidis. Monocular visual odometry in urban environments using an omnidirectional camera. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 2531–2538, 2008. doi:10.1109/IROS.2008.4651205. 28, 34

S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. MIT Press, 2001. URL http://www.probabilistic-robotics.org/. 58, 60

N. Tomatis, I. Nourbakhsh, and R. Siegwart. Hybrid simultaneous localization and map building: a natural integration of topological and metric. *Robotics and Autonomous Systems (RAS)*, 44(1):3–14, 2003. 2, 3, 9, 10, 11, 12, 13, 26, 133

L. Torresani, M. Szummer, and A. Fitzgibbon. Efficient object category recognition using classemes. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 776–789. Springer, 2010. 1, 17, 18, 21, 52

A. Toshev, B. Taskar, and K. Daniilidis. Object detection via boundary structure segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 950–957. IEEE Computer Society, 2010. doi:10.1109/CVPR.2010.5540114. 53, 124

A. Trevor, C. Nieto-Granda, J. Rogers, and H. Christensen. Feature-based mapping with grounded landmark and place labels. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 1451–1456. IEEE, 2010. 53, 122, 123

C. Valgren and A. J. Lilienthal. SIFT, SURF and seasons: Long-term outdoor localization using local features. In *Proceedings of the European Conference on Mobile Robots (ECMR)*, pages 253–258, Sept. 2007. URL http://www.aass.oru.se/Research/Learning/publications/Valgren_and_Lilienthal_2007-ECMR07-SIFT_SURF_and_Seasons.html. 3, 13

C. Valgren, A. J. Lilienthal, and T. Duckett. Incremental topological mapping using omnidirectional vision. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 3441–3447, 2006. doi:10.1109/IROS.2006.282583. 27, 46

C. Valgren, T. Duckett, and A. J. Lilienthal. Incremental spectral clustering and its application to topological mapping. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4283–4288, 2007. doi:10.1109/ROBOT.2007.364138. 27, 46

K. M. Varadarajan and M. Vincze. Functional room detection and modeling using stereo imagery in domestic environments. In *IEEE International Conference on Robotics and Automation (ICRA), Workshop on Semantic Perception, Mapping and Exploration (SPME)*, 2011. 18

S. Vasudevan, S. Gachter, V. Nguyen, and R. Siegwart. Cognitive maps for mobile robots–an object based approach. *Robotics and Autonomous Systems (RAS)*, 55(5):359–371, 2007. 1, 52

A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms, 2008. URL `http://www.vlfeat.org/`. 34, 35

V. Vineet, O. Miksik, M. Lidegaard, M. Niessner, S. Golodetz, V. Prisacariu, O. Kahler, D. Murray, S. Izadi, P. Perez, and P. Torr. Incremental dense semantic stereo fusion for largescale semantic scene reconstruction. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2015. URL `http://www.graphics.stanford.edu/~niessner/vineet2015icra.html`. 19

M. Volkov, G. Rosman, D. Feldman, J. W. Fisher, and D. Rus. Coresets for visual summarization with applications to loop closure. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2015. doi:10.1109/ICRA.2015.7139704. 10, 11, 12

M. Walter, S. Hemachandra, B. Homberg, S. Tellex, and S. Teller. Learning semantic maps

from natural language descriptions. In *Proceedings of Robotics: Science and Systems (RSS)*, Berlin, Germany, June 2013. 17, 19

D. F. Wolf and G. S. Sukhatme. Semantic mapping using mobile robots. *IEEE Transactions on Robotics (T-RO)*, 24(2):245–258, 2008. 53, 122

L. L. Wong, L. P. Kaelbling, and T. Lozano-Pérez. Constructing semantic world models from partial views. In *Proceedings of Robotics: Science and Systems (RSS)*, 2013. URL http://lis.csail.mit.edu/pubs/wong-rssws13.pdf. 19, 52, 122, 123

L. L. Wong, L. P. Kaelbling, and T. Lozano-Prez. Data association for semantic world modeling from partial views. *The International Journal of Robotics Research (IJRR)*, 34(7):1064–1082, 2015. doi:10.1177/0278364914559754. URL http://ijr.sagepub.com/content/34/7/1064.abstract. 19, 52, 122, 123

B. Yao, G. Bradski, and L. Fei-Fei. A codebook-free and annotation-free approach for fine-grained image categorization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Providence, RI, USA, June 2012. 1, 17, 22

A. R. Zamir and M. Shah. Accurate image localization based on google maps street view. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages IV:255–268, 2010. 13, 15, 16

H. Zender, O. M. Mozos, P. Jensfelt, G. Kruijff, and W. Burgard. Conceptual spatial representations for indoor mobile robots. *Robotics and Autonomous Systems (RAS)*, 56 (6):493–502, 2008. 18

K. Zhou, K. M. Varadarajan, A. Richtsfeld, M. Zillich, and M. Vincze. From holistic scene understanding to semantic visual perception: A vision system for mobile robot. In *IEEE International Conference on Robotics and Automation (ICRA), Workshop on Semantic Perception, Mapping and Exploration (SPME)*, 2011. 18

Z. Zivkovic, B. Bakker, and B. Kröse. Hierarchical map building using visual landmarks and geometric constraints. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, Aug. 2005. doi:10.1109/IROS.2005.1544951. 9