January 2001

Enumeration and Motion Planning for Modular Mobile Robots

Sachin Chitta
University of Pennsylvania

James P. Ostrowski
University of Pennsylvania

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-01-08.

Enumeration and Motion Planning for Modular Mobile Robots

Abstract

This report focuses on two different aspects of modular robots, the enumeration of distinct configurations of a modular robot and the generation of gaits for hybrid robots with wheels and legs. Given a particular set of modules from which the robot can be formed, a modular robot made up of these modules can attain a number of different configurations based on the relative attachment of the modules. The distinct configurations possible are enumerated for a locomotion system consisting of a base with multiple ports where wheel or leg modules can be attached.

Given a particular configuration of the modular robot, we would like to generate a set of inputs that would drive the robot from an initial position to a desired position. The method used for this must be applicable to different kinds of modules that may be used for locomotion. The method presented here involves generating a set of constant inputs that will drive a drift-free system from an initial to a final desired position. Simulation results are generated for translation and rotation of the robot and motion along a Lie Bracket direction (sideways motion) for the hybrid mobile robot.

Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-01-08.

Enumeration and Motion Planning for Modular Mobile Robots

CIS Technical Report

Sachin Chitta **James P. Ostrowski**

General Robotics Automation, Sensing and Perception (GRASP) Laboratory, University of Pennsylvania, 3401 Walnut Street, Philadelphia, PA 19104-6228
E-mail:{sachinc,jpo}@grip.cis.upenn.edu

Abstract

This report focuses on two different aspects of modular robots, the enumeration of distinct configurations of a modular robot and the generation of gaits for hybrid robots with wheels and legs. Given a particular set of modules from which the robot can be formed, a modular robot made up of these modules can attain a number of different configurations based on the relative attachment of the modules. The distinct configurations possible are enumerated for a locomotion system consisting of a base with multiple ports where wheel or leg modules can be attached.

Given a particular configuration of the modular robot, we would like to generate a set of inputs that would drive the robot from an initial position to a desired position. The method used for this must be applicable to different kinds of modules that may be used for locomotion. The method presented here involves generating a set of constant inputs that will drive a drift-free system from an initial to a final desired position. Simulation results are generated for translation and rotation of the robot and motion along a Lie Bracket direction (sideways motion) for the hybrid mobile robot.

1 Introduction

Modularity in robotics has been widely studied in recent years. Modular robots are robots made from a set of modules. Modularity in any system implies that any individual part of the system is one of a set of separate parts which, when combined, form a complete whole system. Modular robots are usually classified into two categories

1. Manually re-configurable or re-configurable robots – These are robots where new or additional modular parts can be attached to a fixed or moving base. The new parts are attached manually, i.e. by an external agent.
2. Self re-configurable robots – These are robots with the ability to reconfigure using their own power.

1.1 Locomotion of modular robots

Locomotion of modular robots has been widely studied. Most of the work has focused on developing control or motion algorithms for robots made up of a particular set of modules. In [1], Yim describes various behavioral modes of locomotion that allow the Polypod, a multi-dof self re-configurable robot, to carry out different locomotion tasks. The robot carries out a sequence of motion in which each joint is allowed to run under a particular control algorithm until a certain trigger condition is met. A master controller then changes the control or behavioral mode for the joint. Thus, any motion is composed of a set of triggers and behavioral modes. An example of the behavioral mode is a spring mode where the joint behaves like a spring.

In [2], Rus, et. al. describe the Inchworm, a robot capable of manipulation and locomotion tasks. The approach involves defining a series of steps in each of which the robot carries out a specific manipulation or locomotion task. In [3], Rus, et. al. present the concept of a robotic molecule many of which can be attached together to form a aggregate structure capable of locomotion. Chirikjian, et. al.[4] describe a self re-configurable robot made up of a number of 2D hexagonal modules. Locomotion is achieved by such robots using a sequence of steps wherein individual modules move relative to the robot and attach themselves at a new position on the robot. This allows the whole robot to achieve a net planar motion. Murata, et. al.[5] have developed a 3d re-configurable robot capable of motion by attaching itself to similar modules. A 2d planar robot [6] made up of homogenous units moves by alternate repulsive and attractive forces generated by an electromagnet.

The aim of our research is to develop a general framework for robot locomotion which can be widely applied. The approach in [1] is very specific to the particular robot chosen. In [2], the locomotion and manipulation task is broken into a sequence of specific tasks which are described qualitatively. The exact manner in which a particular task can be carried out is not specified. Our approach aims to be general enough that it can handle a variety of problems and yet specific enough to generate a particular solution for each different problem.

1.2 Scope of report

This report focuses on two different aspects of modular robots, the enumeration of distinct configurations of a modular robot and the generation of gaits for hybrid robots with wheels and legs. Given a particular set of modules from which the robot can be formed, a modular robot made up of these modules can attain a number of different configurations based on the relative attachment of the modules. An algorithm for enumerating the number of non-isomorphic or distinct configurations of such a modular robot was presented by Chen and Burdick in [7] which is discussed in this report. The method is applied to the specific case of modular robots with different numbers of wheels and legs.

Given a particular configuration of the modular robot, we would like to generate a set of inputs that would drive the robot from an initial position to a desired position. The method used for this must be applicable to different kinds of modules that may be used for locomotion. The method presented here is used with two types of modules, legs and wheels to generate control inputs for a hybrid modular robot. The method was developed for smooth systems in [8] and extended to include the concept of stratified systems which represent legged robots in [9].

1.3 Outline of report

A method of representing modular robots in terms of graph and matrix representation is discussed in Section 2. A set of modules used for locomotion is presented. In Sections 3 and 4, the enumeration algorithm for modular robots is discussed and its application to locomotion is presented using a few examples. Section 5 presents the motion planning method for hybrid modular robots. The method used is first described for smooth systems. Its extension to stratified systems is then presented. Our approach to the problem using the description of the motion of the system in terms of shape and body inputs is then described. The method is then applied to two different robots in Section 6, a four legged robot and a robot with two legs and a pair of wheels and simulation results are presented for different motions of the two robots. Section 7 presents the conclusions and discusses possible extensions of this method.

2 Graph and matrix representation of modular robots

Modular robots can be made up heterogeneous or homogeneous modules. A representation scheme for modular robots must accurately represent the heterogeneous or different kinds of modules and the interconnection between them. Given a representation, it should be possible to reconstruct the actual physical robot uniquely. Representation schemes are particularly useful for developing motion planning or reconfiguration algorithms wherein a description of the initial and target shape may be required. They also help to store information about modular robots in a compact form. In [7], Chen and Burdick present a general method of representing modular robots. The method uses a graph structure and a corresponding matrix representation for representing such robots.

2.1 Modules for locomotion

Two kinds of modules are defined in [7]. They are based on the most general kinds of modules present in most robots. The modules are broadly classified as link modules and joint modules. Link modules can come in various shapes and have ports where joint modules can be attached. Each link module can have a multiple number of ports. Joint modules attach two link modules to each other and allow relative motion between the two modules. Examples of link modules are the prism module with 10 ports and the cubical module with 6 ports shown in Figure 8. The ports of each module are numbered as shown in the figure. Examples of joint modules include revolute, cylindrical, prismatic and helical modules. Modules defined for locomotion generally include a mobile base module with ports to which other modules can be attached. The 4 port prism module shown in figure 1 is one such module. Two other kinds of modules are also defined for locomotion tasks. They are leg and wheel modules as shown in Figure 2. Leg modules are made up of a series of links that can be attached at one end to a port on the base of the robot. A set of attributes or properties can be specified for such modules that describe the workspace for the leg and its parameters. For a wheel, the main attribute would be its radius.

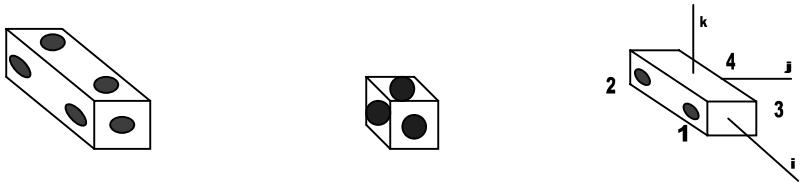


Figure 1: Prism(L) and Cubic(B) modules with 10 and 6 ports respectively and a four port prism module

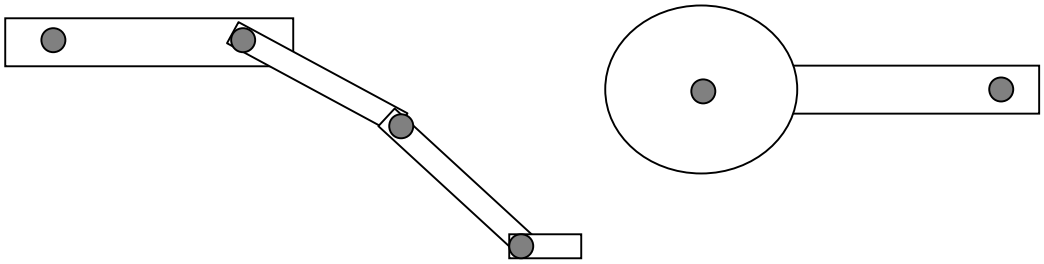


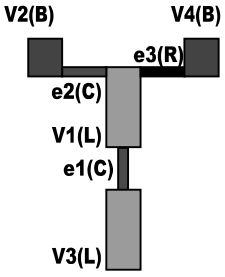
Figure 2: Leg and wheel modules attached to a base

2.2 Graph and Matrix representation of modular robots

Kinematic chains of links and joint are often represented as graphs. By replacing the joints in a chain by edges and the links by vertices, a kinematic graph representation can be created. A graph essentially consists of a set of vertices and edges. Each edge links two different vertices. In a labeled graph, the vertices and edges are assigned labels. If the

vertices and edges are of different kinds, e.g. in a heterogeneous modular robot, a specialized graph structure can be defined. Consider a labeled graph with a set of vertices $V = \{v_1, v_2 \ldots\}$ and edges $E = \{e_1, e_2 \ldots\}$. If $\boldsymbol{V}$ and $\boldsymbol{E}$ are the set of types of vertices and edges, then an injection mapping $f_v: V \rightarrow \boldsymbol{V}$ and $f_e: E \rightarrow \boldsymbol{E}$ assigns a particular vertex and edge type to each of the vertices and edges of the graph respectively. In case of the representation scheme for modular robots, the types of vertices and edges are the types of link and joint modules respectively.

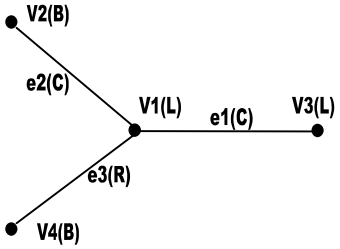
The graph representation is often converted into a matrix representation for convenience. The type of matrix representation used here is known as a vertex-edge incidence matrix. The element m_{ij} of this matrix has value 1 if edge e_j is incident on vertex v_i, otherwise it has value 0. Thus, each row of a matrix representation contains information about one vertex of the graph while each column contains information about a edge of the graph. To represent specialized graphs an additional row or column is added to the matrix to indicate the type of the vertex and edge respectively. The resulting matrix representation for a specialized graph is known as an extended incidence matrix. In representing modular robots, the non-zero elements of this matrix are replaced by the port numbers of the links where the corresponding joints are attached thus giving an assembly incidence matrix. Figure 3 shows an example of a hybrid modular robot and extended incidence and assembly incidence matrices for the same. This representation scheme is general enough to be used with different kinds of modules.



A hybrid modular robot

$$M(G) = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ \\ \\ \end{array} \begin{array}{c} \begin{array}{cccc} e_1 & e_2 & e_3 & \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & L \\ 0 & 1 & 0 & B \\ 1 & 0 & 0 & L \\ 0 & 0 & 1 & B \\ C & C & R & 0 \end{pmatrix} \end{array}$$

Incidence Matrix



Specialized graph representation

$$A(G) = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ \\ \\ \end{array} \begin{array}{c} \begin{array}{cccc} e_1 & e_2 & e_3 & \end{array} \\ \begin{pmatrix} 10 & 5 & 1 & L \\ 0 & 1 & 0 & B \\ 9 & 0 & 0 & L \\ 0 & 0 & 2 & B \\ C & C & R & 0 \end{pmatrix} \end{array}$$

Assembly Incidence Matrix

Figure 3: A hybrid modular robot and its graph and matrix representations

3 Enumerating distinct robot configurations - Pattern Enumeration

The problem of enumerating all possible distinct arrangements of modular systems given a set of modules is important. In [7], a method for such an enumeration is presented. The method is based on testing for two conditions given a pair of modular robots –the equivalence of individual link assembly patterns and the isomorphism of the underlying graph structure. These methods are now described in more detail in sections 3 and 4 respectively after the introduction of a few concepts used in them.

3.1 Symmetric rotations

A symmetric rotation for a link is a rotation about a body axis after which the original and rotated links cannot be distinguished. Symmetric rotations of a module can be identified with certain features of the module. Here, the port numbers on a link are used to identify a symmetric rotation as a permutation on the set of port numbers. For example consider a prism link rotated through 90^{o} about the z axis. As can be seen from figure 4, port 1 now occupies the position where port 3 originally was while port 3 occupies the space where port 5 originally was and so on. Thus, the symmetric rotation can be represented as

$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 3 & 4 & 5 & 6 & 7 & 8 & 1 & 2 & 9 & 10 \end{pmatrix}$$

Each link has a set of possible symmetric rotations associated with it which are represented as permutations on the set of port numbers for the link. Let PORT = {1,2,3,4,5,6,7,8,9,10} denote the set of port numbers for the prism link. Let π denote a permutation on PORT where $\pi \in \boldsymbol{S}$. $\boldsymbol{S}$ is the set of permutations corresponding to symmetric rotations of the prism link. Table 1 lists all the symmetric rotations and corresponding permutations for a four port rectangular base shown in figure 8.

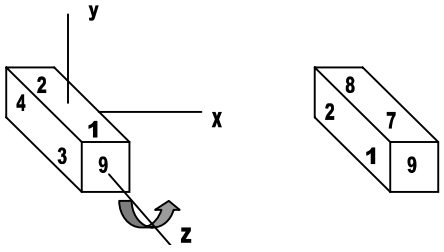


Figure 4: An example of a symmetric rotation for a prism link

Permutations on Port→ Rotation	1	2	3	4	Type
Identity	1	2	3	4	{4,0,0,0}
About *x* (180^0)	4	3	2	1	{0,2,0,0}
About *y* (180^0)	2	1	4	3	{0,2,0,0}
About z (180^0)	3	4	1	2	{0,2,0,0}

Table 1: The symmetric rotation group for a four port prism link

3.2 Assembly patterns for a single link

An assembly pattern $\boldsymbol{f}$ for a single link is a unique assignment of joints from a set (denoted by ATT) of possible joints to each port of the link. Thus, $\boldsymbol{f}$: PORT→ATT. For example, ATT = {R,P,H,C,0} denoting a rotary, prismatic, helical and cylindrical joint respectively. The set of possible joints includes a null joint indicating that no joint has been attached to that particular port of the link. Let m = |ATT| and n = |PORT|. Then the number of assembly patterns possible is m^n. Let $\boldsymbol{F}$ denote the set of all possible assembly patterns. Two assembly patterns $\boldsymbol{f}_i$ and $\boldsymbol{f}_j$: PORT→ATT are equivalent iff $\exists\ \pi \in \boldsymbol{S}$ such that

$$\boldsymbol{f}_i = \boldsymbol{f}_j \circ \pi$$

Since the cubic and prism links are symmetric, two assembly patterns may look alike after a symmetric rotation of the link. Consider the assembly patterns $\boldsymbol{f}_a$ and $\boldsymbol{f}_b$ shown in Figure 5. If π is a rotation of the prism about the z axis by 90°,

$$\boldsymbol{f}_a(1) = \boldsymbol{f}_b \circ \pi(1) = \boldsymbol{f}_b(3) = \text{H}$$
$$\boldsymbol{f}_a(2) = \boldsymbol{f}_b \circ \pi(2) = \boldsymbol{f}_b(4) = \text{R}$$
$$\boldsymbol{f}_a(9) = \boldsymbol{f}_b \circ \pi(9) = \boldsymbol{f}_b(9) = \text{C}$$
$$\boldsymbol{f}_a(i) = \boldsymbol{f}_b \circ \pi(i) = 0 \text{ for all other } i$$

Thus, both the assembly patterns will perform identically in a modular robot.

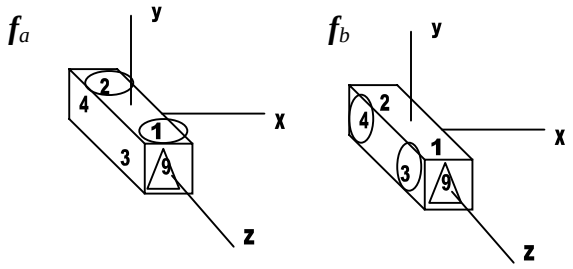


Figure 5: Assembly patterns for a prism link with two rotary joints (indicated by circles) and one cylindrical joint (indicated by a triangle).

3.3 Enumeration of equivalence classes using Polya's theorem

The set of symmetric rotations divides the set of assembly patterns ***F*** into disjoint subsets called equivalence classes. The assembly patterns belonging to a particular equivalence class are all related by a symmetric rotation. Thus, the problem of finding the number of distinct assembly patterns is converted into one of finding the number of such equivalence classes. Polya's theorem is used to find this number. To use Polya's theorem, we first need to establish some notation.

A permutation π operating on PORT splits the index set into cycles. Length of a cycle = m if $\pi^m(s) = s \in$ PORT and s, π (s),…., $\pi^{m-1}(s) \in$ PORT .The type of a permutation, type(π) = $\{b_1,b_2,b_3,b_4,\ldots..,b_n\}$ where b_i=no. of cycles of length i. For example, rotation through 90^o splits PORT into four cycles {1,3,5,7}, {2,4,6,8}, {9}, {10} and hence type (π) = {2,0,0,2,0,0,0,0,0,0}.We now define the cycle index of a permutation group ***S*** to be a polynomial in dummy variables $x_1,x_2,\ldots x_n$ given as:

$$P_S(x_1,x_2,..,x_n) = \frac{1}{|S|}\sum_{\pi\in S} x_1^{b_1} x_2^{b_2} \ldots. x_n^{b_n}$$

where type(π) = $\{b_1,b_2,b_3,b_4,\ldots..,b_n\}$.

Now, consider J = set of joints$\{J_1,J_2\ldots J_m\}$, e.g.{R,P,0}. Assign a dummy variable y_i to J_i, i=,1,2,3…..,m=$|J|$. In the cycle index polynomial substitute

$$x_k = \sum_{i=1}^{|J|} y_i^k$$

Then the coefficient of

$$y_1^{d_1}\, y_2^{d_2} \ldots..\ \ y_m^{d_m}$$

gives the number of distinct patterns or equivalence classes for a single link with d_1 joints of type J_1 , …., d_m joints of type J_m . Having found the number of distinct equivalence classes, an algorithm is now required for enumerating the distinct classes. The algorithm for this purpose is essentially a brute force algorithm and is presented below.

Algorithm ***PatternEnumerate***

Input to Algorithm

Set of Assembly patterns (***F***)

Symmetric rotation group (***S***)

Algorithm

Step 1 : Queue = ***F*** ;

Step 2 : v = First(Queue); Queue = Rest(***F***); Dpattern = {};

Step 3 : Vpattern = { v o π , $\pi \in$ ***S*** **}**

Step 4 : $\forall f_i \in$ Queue, if $f_i \in$ Vpattern then Queue = Queue - {f_i }

Step 5 : Dpattern = Dpattern + {v}

Step 6 : If Queue = {k}, Dpattern = Dpattern + {k}

Step 7 : Repeat from Step 2.

Output of Algorithm

The set {Dpattern} of distinct assembly patterns under *S.*

END

The set Vpattern is the set of all patterns found by applying a symmetric rotation to the assembly pattern v. The rest of the patterns are then tested against this set and if any of them is found to belong to the set, it is discarded from the queue. Thus, the algorithm gives a set of equivalence classes or distinct assembly patterns.

4 Distinct modular robot configurations

4.1 Graph isomorphism

Having found the distinct assembly patterns for a single link, the problem now is to find distinct modular robot assemblies. The concept of graph isomorphisms is useful in solving this problem. Two graphs G_1 and G_2 are said to be isomorphic to each other if their incidence matrices M_1 and M_2 are related to each other by row or column permutations. If γ_{12} is the isomorphism between the two graphs then

$$M_{\gamma_{12}}(G_1) = M(G_2)$$

A graph may be isomorphic to itself in which case

$$M_{\gamma}(G_1) = M(G_2)$$

When testing for pattern equivalence on specialized graphs it is important to take into account the type of edge or vertex.

4.2 Equivalence of two modular robot configurations

4.2.1 Topological equivalence

The problem of enumerating distinct modular robot configurations can now be addressed. Given two modular robot assembly configurations, their graph representations G_1 and G_2

and their assembly incidence matrices (AIMs) A(G_1) and A(G_2), the first step is to test for the topological equivalence of the two robot configurations.

Two robot configurations and their AIMs, A(G_1) and A(G_2), are topologically equivalent iff G_1 and G_2 are isomorphic.

Physically, an isomorphism represents a re-labeling of the links and edges of a modular robot and does not affect the actual physical structure of the robot. For specialized graphs that represent hybrid robots, the test for topological equivalence is the same except that the type of link must be taken into account when carrying out the test for isomorphism.

4.2.2 Pattern equivalence

If *A(G_1)* and *A(G_2)* are topologically equivalent, let γ_{12} denote the isomorphism from G_1 to G_2. Let $w_i^1 = \{a_{i1}^1, a_{i2}^1 \ldots.\}$ and $w_i^2 = \{a_{i1}^2, a_{i2}^2 \ldots.\}$ be i^{th} row vectors of $A\gamma_{12}(G_1)$ and $A(G_2)$. w_i^1 and w_i^2 will have non-zero elements in the same position in the row. w_i^1 and w_i^2 are pattern equivalent iff $\exists\, \pi : \text{PORT} \rightarrow \text{PORT}$ such that $\forall\, a_{ij}^1 \in \text{PORT}$, $\pi(a_{ij}^1) = a_{ij}^2$, ie iff they are related by a symmetric rotation.

When checking for pattern equivalence, it is important to take into account automorphisms of the graph to itself. It is necessary to compare all automorphisms as well when checking for pattern equivalence of these two rows. For specialized graphs which represent hybrid robots the test for pattern equivalence is the same except that the symmetric rotation group for each different type of link will be different.

4.3 The enumeration algorithm

The enumeration algorithm to enumerate and list out distinct modular robot configurations follows.

Step 1: Generate non-isomorphic trees $\{G_i\}$ and corresponding incidence matrices for a given number of vertices. Several computer algorithms are present which can be used to carry out this step.
Step 2: Find the automorphism groups for each of the trees.
Step 3: The algorithm PatternEnumerate is used to find distinct assignments from Link (the set of all types of links) and Joint (the set of all types of joints) to vertices and edges of G_i. This problem is similar to the problem of finding distinct assembly patterns for a link except that the set PORT is now replaced by the set VERTEX or EDGE and the set ATT is now replaced by JOINT or LINK. Construct non-isomorphic specialized graphs G_i and corresponding extended incidence matrices for each of these assignments.
Step 4: Find the automorphism groups for each non-isomorphic specialized graph G_i.
Step 5: For every G_i, generate distinct assembly patterns for every link. This is done by using algorithm, PatternEnumerate, since the number of joints on each link is known from the rows of the incidence matrix. All the labeled joints are treated as different since

they will be attached to different links. Combine all pattern inequivalent row vectors for every link to construct inequivalent AIMs.
Step 6: Use automorphism groups to eliminate equivalent AIMs.
Step 7: Repeat Step 5 for every G_i .

4.4 An example

Consider as an example a modular robot consisting of 3 links-a prism link with four ports as shown in Figure 10 and two cubic links and 2 rotary joints. In the first step, we can generate the non-isomorphic graphs for the modular robots. There is only one such graphs possible, a serially connected graph as shown in Figure 12. In Step 3, two non-isomorphic specialized graphs are constructed as shown in Figure 12. G_2 is symmetric and hence has an automorphism which makes the graph isomorphic to itself.

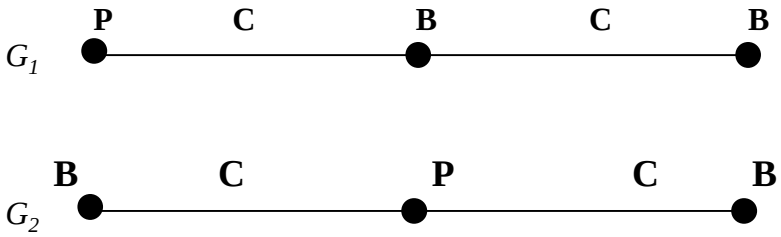


Figure 6: Two non-isomorphic specialized graphs. B-cubic link, P –Prism link with four ports and C cylindrical joint

For graph G_1 , the cubic link with one joint attached to it has only one distinct assembly pattern. The cubic link with two joints attached to it has two distinct assembly patterns possible. The prism link with four ports and one joint attached to it has one distinct assembly pattern. This can be confirmed using Polya's theorem. For the prism link with four ports, the set of symmetric rotations ***S*** has 4 elements from Table 2. The set of Joints ***J*** ={C,C,0}. The cycle index polynomial will be

$$P_S(x_1, x_2, .., x_n) = \frac{1}{|S|}\{(x_1{}^4) + (x_2{}^2) + (x_2{}^2) + (x_2{}^2)\}$$

The set of Joints ***J*** ={C,C,0}. Assign y_1 to the first cylindrical joint, y_2 to the second cylindrical joint and y_3 to the null joint. Then, substitute in the cycle index,

$$x_k = \sum_{i=1}^{3} y_i^k$$

to get the polynomial

$$P_S = \frac{1}{4}(y_1 + y_2 + y_3)^4 + \frac{3}{4}(y_1{}^2 + y_2{}^2 + y_3{}^2)^2$$

The number of distinct assembly patterns with a single joint is then given by the coefficient of $y_1 y_3^3$ which is 1 as expected. Thus there are a total of two distinct robot configurations based on the graph G_1.
For graph G_2 , the two cubic links each have one joint attached to them and thus have one distinct assembly pattern. The prism link will have 3 distinct assembly patterns (3 is the coefficient of $y_1 y_2 y_3^2$ in P_S) . Thus there are a total of 3 distinct modular robot configurations based on the graph G_2. The total number of modular robot configurations is thus 3 + 2=5. They are shown in Figure 7.

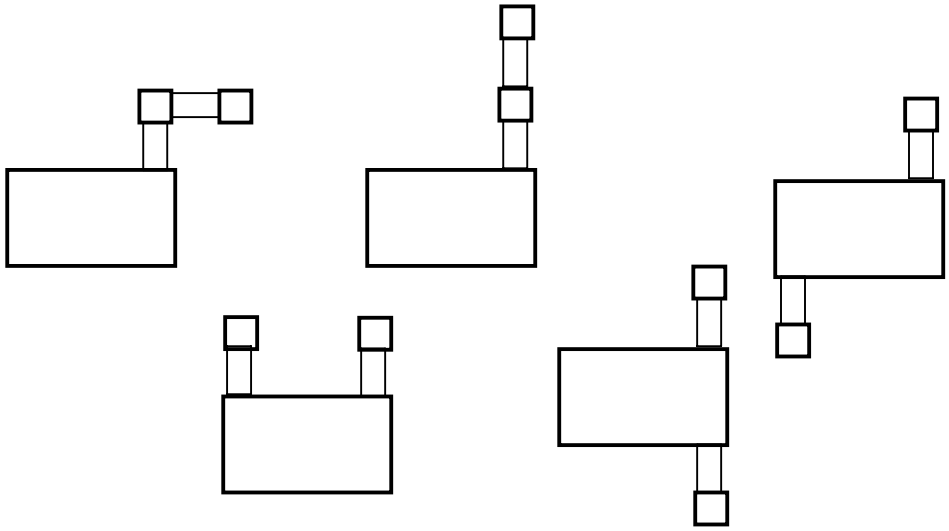


Figure 7: Five distinct modular robot configurations. The bigger rectangle is the four port base, the small squares represent the cubic modules and the smaller rectangles represent the cylindrical joints.

4.5 Example - Application to locomotion

Consider a second example of a mobile robot with a 6 port base and 3 legs and 3 wheels. In the case of a mobile robot of this form, only one module can be attached to each port of the base link, either a wheel or a leg module. The 6 port base and its set of symmetric rotations is shown in Figure 8. This problem can be solved by the application of Polya's theorem.

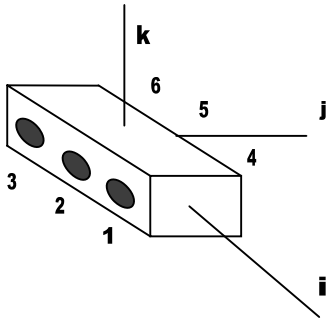


Figure 8: 6 port prism link

Permutations on Port→ / Rotation	1	2	3	4	5	6	Type
Identity	1	2	3	4	5	6	{6,0,0,0,0,0}
About *x* (180^0)	4	5	6	1	2	3	{0,3,0,0,0,0}
About *y* (180^0)	3	2	1	6	5	4	{2,2,0,0,0,0}
About z (180^0)	6	5	4	3	2	1	{0,3,0,0,0,0}

Table 2: The symmetric rotation group
for a six-port prism link

The set of joints is now the set of modules ={Leg, Wheel, Null}. The cycle index polynomial is now

$$P_S = \frac{1}{4}\{(y_1 + y_2 + y_3)^6 + (y_1^2 + y_2^2 + y_3^2)^3 + (y_1 + y_2 + y_3)^2(y_1^2 + y_2^2 + y_3^2)^2 + (y_1^2 + y_2^2 + y_3^2)^3\}$$

The number of distinct configurations for a robot with 3 legs and 3 wheels is given by the coefficient of $y_1^3 y_2^3$ in the above polynomial which is 6. These 6 configurations are shown in Figure 9.

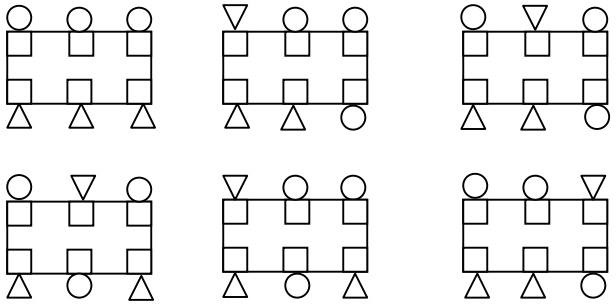


Figure 9: Distinct configurations for a robot with three legs and three wheels, the triangle represents the legs and the circles represent the wheels.

Thus, the algorithm can be successfully applied to determine the distinct configurations for a mobile robot. We have only presented the enumeration of distinct configurations for a robot with three legs and three wheels. The distinct configurations possible for a robot

with all 6 ports filled, i.e. there is either a leg or a wheel module on each of the ports, is given in Table 3. Using the above polynomial, the number of distinct configuration with different number of ports filled can e easily found. Given the distinct configuration, we now address the problem of locomotion for the robot, in particular the generation of a set of inputs that will take the robot from an initial position to a final desired position.

No. of wheels	0	1	2	3	4	5	6
No. of legs	6	5	4	3	2	1	0
No. of distinct configurations	1	2	6	6	6	2	1

Table 3: Number of distinct configurations for a 6 port base with different numbers of legs and wheels

5 Steering for hybrid robots

In implementing motion planning for hybrid robots, the task is to determine a set of inputs that will steer the robot from an initial state to a final state. In this section, a method for motion planning for kinematic, drift-free hybrid robots is presented. The method can be applied to any robot with legs and wheels. The method is independent of the number of legs and wheels.

5.1 The basic method

The dynamics of a hybrid robot can in general be described as

$$\dot{x} = f(x) + g(x)u,\ x \in \Re^n, u \in \Re^m. \quad (1)$$

Thus, the robot has m inputs and n states. If the term $f(x) = 0$, the system is known as a ***drift-free*** system. Henceforth, we will only consider drift-free systems. A method for finding control inputs for drift-free, smooth systems was first presented by Lafferriere and Sussmann in [2]. Smooth systems are systems where the equations of motion remain the same throughout the motion of the robot. An example of such a system is a wheeled mobile robot. The wheels of the robot are always in contact with the ground. Such a mobile robot has a set of allowed directions in which it can move. The robot can nevertheless move in other forbidden directions by a combination of motions along the allowed directions.

The approach involves the calculation of a set of vector fields that span the configuration space of the robot with corresponding inputs along which the robot can move. An ***extended system*** is formed using this set of vector fields and corresponding inputs. Some of these inputs may not correspond to real inputs for the robot and hence these inputs are known as ***fictitious inputs***. The motion of the robot corresponding to these fictitious inputs can be achieved by a sequence of real inputs. This has the net effect of a motion along a direction for which there is no real input. The method generates a sequence of constant control inputs that drive the robot to its desired state.

5.2 Inputs to a mobile robot

To represent the dynamics of a mobile robot, it is often useful to formulate the equations of motion in terms of a set of general inputs. This approach would be particularly useful in representing locomotion modular robots as it would be easy represent the effect of addition of new modular components. There can be two kinds of general inputs for a robot. Shape inputs represent local changes in the shape of the robot, i.e. changes in joint angles, position of the end effector, etc. Body inputs specify the overall motion of the body in terms of its position or relative orientation with respect to a coordinate reference frame.

The configuration space for a robot usually includes the position of its center of mass, its orientation and the position of the shape variables for the robot which could include the joint angles, etc. If x represents the configuration space of a robot, its motion can be represented as

$$\dot{x} = C(x)u.$$

which can be further split up as

$$\dot{x} = A(x)\xi + B(x)u^r.$$

Here u is the set of inputs to the system. ξ is the set of body or non-shape inputs while u^r is the set of shape inputs to the system. For stratified systems which represent legged robots, the set of shape inputs may change for different strata. The concept of stratified systems is present later in this report.

The set of body inputs can be further represented in terms of allowable body motions. In particular for robots with wheels, there is a set of allowable directions in which the robot is allowed to move. This can also be represented as a set of constraints on the motion of the robot. The body inputs can then often be represented in terms of a set of allowable inputs. For a pair of wheels, this set of inputs can be thought of as a drive velocity along a direction perpendicular to the axis of the wheels and the steering angle which is the angle the drive velocity makes with respect to a body fixed coordinate system. Thus

$$\xi = f(x)u^w.$$

where u^w is the set of allowable velocities for the robot. For a wheeled robot (Figure), we have

$$\xi = \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \\ \dot{\theta} \end{bmatrix} = f(x)u^w = \begin{bmatrix} \cos(\phi) & 0 \\ \sin(\phi) & l \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}$$

where $\dot{x}_b$ and $\dot{y}_b$ are body velocities. For a legged robot, this would just be

$$\xi = \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \\ \dot{\theta} \end{bmatrix}$$

Given a hybrid robot , the formulation above makes it easy to write down the equations of motion for the combined system as will be shown for a robot with legs and wheels in section 6.2.1.

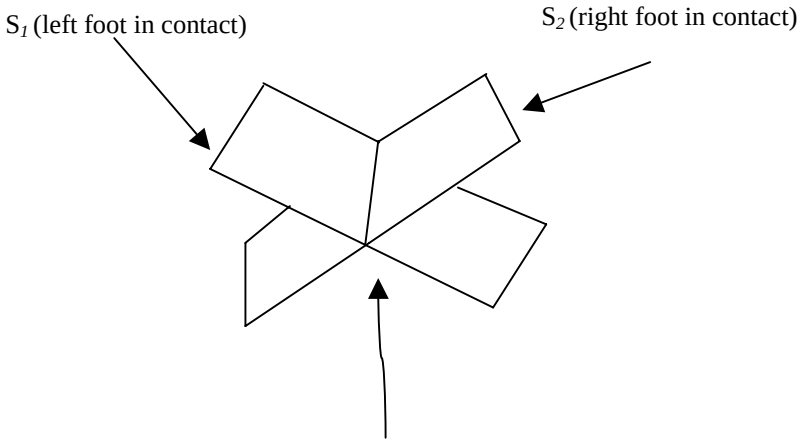
5.3 Stratified configuration spaces

The approach is extended to legged robots in [9] where Goodwine and Burdick present the concept of stratified sets. The legs of a legged robot continuously make and break contact with the surface over which the robot is moving. Thus, for legged systems the equations of motion are not always the same. The equations of motion of a legged robot change continuously with the contact of different feet and depend on the type and number of legs in contact.

The configuration space for a legged robot can be considered made up of a number of strata. Each stratum describes the space over which the states of the system evolve with one or more legs in contact. Consider a kinematic quadruped robot. The configuration space for the robot could include the position and orientation of the robot and the position and the joint angles for each of its legs. With three legs in contact with the ground, the robot is subject to a set of constraints that restrict its motion. Thus, the

motion of each of the legs on the ground would be related to the body velocity and rotation of the robot. The free leg in the air would not be subject to the same constraints and can thus be moved around freely off the ground. With all four legs in contact, all the legs will be constrained.

Stratified configuration space for a biped



With one foot in contact, the system would evolve on a submanifold of dimension less than that of the configuration space. Each additional foot in contact would further reduce the dimension of the submanifold over which the system evolves. Each of these submanifolds over which the system evolves when one or more of the feet are in contact is thus a stratum. The equations of motion for the robot are smooth on a particular stratum. Discontinuities arise only when the robot moves from one stratum to another. Chen and Burdick represent this concept of stratified sets as shown in the Figure above for a biped (two legged robot).

Let M denote the entire robot's configuration space. Let $S_{ijkl..p}$ denote the submanifold over which the system evolves with the i^{th}, j^{th}, k^{th},l^{th} …, p^{th} legs in contact with the ground. The submanifold with more legs in contact is said to be lower than the submanifold with lower number of legs in contact. The bottom stratum is the submanifold with all the legs in contact with the ground.

The method of [8] can be applied to legged robots only if the system evolves on a single manifold. This is achieved by allowing the system to evolve on a set of vector fields from all the different strata. Each of these vector fields is assumed to hold on the bottom stratum and the method of [8] can then be applied on the bottom stratum. The resulting set of input is then implemented on the appropriate stratum. The next section discusses this method in greater detail.

5.4 Steering for drift-free system

Consider a drift free system given as

$$\dot{x} = g_1 u^1 + g_2 u^2 + ... + g_m u^m, x \in \Re^n, g_i \in \Re^n . \qquad (2)$$

In general m<n, i.e. the system does not have enough inputs to directly drive each of the states to the desired values. Therefore, an additional set of vector fields is added to the above system to span the entire configuration space. An extended system based on the system given above is defined as

$$\dot{x} = b_1 v^1 + b_2 v^2 + ... + b_m v^m + b_{m+1} v^{m+1} + ... + b_s v^s. \qquad (3)$$

where

$$\dim\ (\text{span}\ \ \{b_1, b_2, ..., b_m, b_{m+1} ....., b_s\}) = \dim\ \ (M). \qquad (4)$$

The b_i's are chosen from the Phillip Hall basis (Appendix 1) which is a basis for the Lie algebra defined by the set of vector fields $\{g_i(x)\}$. Here $b_i = g_i, i = 1 \text{ to m}$ and the $b_{m+1}, ..., b_s$ correspond to higher order Lie brackets (A description of Lie brackets is given in Appendix 1) chosen from the Philip Hall basis. The v_i's are called ***fictitious inputs*** to the system since they may not correspond to the real inputs to the system. Let x_i and x_d denote the initial and desired states of the system respectively. Define a curve $\gamma\ (t)$ going from x_i to x_d. The curve could be a straight line although any other curve would also work. The v_i's can be solved by solving a set of simultaneous equations formed from equation (1). The solution involves taking a pseudo inverse when *s* < *n*. The actual inputs can be determined from the fictitious inputs in the following manner:

Determine the b_i's which span the configuration space M from the Phillip Hall basis for the Lie algebra generated by $g_1, g_{2,}, ..., g_m$. Then all the flows of equation (1) can be represented as

$$S\ (\text{t}) = e^{h_s(t)b_s} e^{h_{s-1}(t)b_{s-1}} .....e^{h_2(t)b_2} e^{h_1(t)b_1}. \qquad (5)$$

The h_i's are called the (backward) Philip Hall coordinates. Differentiating the above equation we get

$$\dot{S}\ (\text{t}) = \sum_{j=1}^{s} S(t) e^{-h_1(t)b_1} ....e^{-h_{j-1}(t)b_{j-1}} b_j \dot{h}_j e^{h_{j-1}(t)b_{j-1}} ...e^{h_1(t)b_1}. \qquad (6)$$

This simplifies to

$$\dot{S}(t) = \sum_{j=1}^{s} S(t) Ad_{e^{-h_1 b_1} .....e^{-h_{j-1} b_{j-1}}} b_j \dot{h}_j\ (\text{t}). \qquad (7)$$

where the adjoint mapping is defined as $Ad_{e^{-h_i b_i}} b_j = e^{-h_i b_i} b_j e^{h_i b_i}$.

Further, we can represent

$$Ad_{e^{-h_1 b_1} .....e^{-h_{j-1} b_{j-1}}} b_j \dot{h}_j\ (\text{t}) = (\sum_{k=1}^{s} p_{j,k}(h)\, b_k) \dot{h}_j. \quad (8)$$

where $p_{j,k}(h)$ is a polynomial in the h_i's.

S(t) also satisfies another differential equation given by

$$\dot{S}\ (\text{t}) = S(t)(b_1 v_1 + ... + b_s v_s); S(0) = 1 \qquad (9)$$

Using equations (7),(8) and (9) and equating the coefficients of the b_i's , we get a set of differential equations for the h_i's,

$$\dot{h} = A(h)v,\ h(0)\ = 0. \qquad (10)$$

These equations specify the evolution of the Philip Hall basis in response to the evolution of the fictitious inputs. The determination of the actual inputs is easier to do given the forward Philip Hall coordinates which can be determined from the backward Philip Hall coordinates by a simple algebraic transformation. However, this transformation can be avoided by assuming that the system is nilpotent of order 2 or can be approximated as a system of nilpotent order 2. This is a reasonable assumption in most cases. A higher order approximation will lead to smaller error. However, usually a higher order Lie bracket direction is harder to achieve physically. Thus, backward Philip Hall coordinates are used henceforth. The conversion to the real inputs is illustrated with a simple example.

5.4.1 Example

Suppose the required backward Philip Hall coordinates have been found and the resultant flow looks like $e^{h_3[g_1,g_2]}e^{h_2g_2}e^{h_1g_1}$. The flow $e^{h_ig_i}$ can be achieved by setting the input $u_i = h_i$ for a time of 1 second. Let a_i denote the required input to achieve the flow e^{g_i}. Then, the flow $e^{h_3[g_1,g_2]}$ is achieved by a sequence of inputs $\sqrt{h_3}a_1\#\sqrt{h_3}a_2\#-\sqrt{h_3}a_1\#-\sqrt{h_3}a_2$ where the # indicates a concatenation. The flow $e^{h_3[g_1,g_2]}e^{h_2g_2}e^{h_1g_1}$ thus corresponds to a sequence of inputs given as

$$u_1 = \sqrt{h_3} \text{ for t} = 0 \text{ to } 1 \text{ s}$$
$$u_2 = \sqrt{h_3} \text{ for t} = 1 \text{ to } 2 \text{ s}$$
$$u_1 = -\sqrt{h_3} \text{ for t} = 2 \text{ to } 3 \text{ s}$$
$$u_1 = -\sqrt{h_3} \text{ for t} = 3 \text{ to } 4 \text{ s}$$
$$u_2 = h_2 \text{ for t} = 4 \text{ to } 5 \text{ s}$$
$$u_1 = h_1 \text{ for t} = 5 \text{ to } 6 \text{ s}$$

Using, the Campbell-Baker-Hausdorff formula it can be proved that the sequence of inputs $\sqrt{h_3}a_1\#\sqrt{h_3}a_2\#-\sqrt{h_3}a_1\#-\sqrt{h_3}a_2$ actually gives rise to $e^{h_3[g_1,g_2]}e^{\alpha[g_1,[g_1,g_2]]}e^{\alpha[g_2,[g_1,g_2]]}$ where $\alpha = \frac{1}{2}h_3^{\frac{3}{2}}$. However, since the system has been assumed to be nilpotent of order 2, the third order bracket term is actually zero. In some cases, this term has to be taken into account since it actually represents a flow along a vector field belonging to $b_1, b_2, ..., b_m, b_{m+1}, ..., b_s$. Thus, a higher order approximation may sometimes yield better results as is shown in the solution for a wheel- legged robot presented later.

5.5 Extension to Stratified Configurations spaces

When extended to stratified spaces, the method remains essentially the same except in the manner in which the b_i's are chosen. For stratified systems, the equations of motion differ for different strata. Consider two strata S1 and S2 with the equations of motion on each strata defined as

$$\dot{x} = g_{S_1,1}u^1 + g_{S_1,2}u^2 + ... + g_{S_1,m}u^m, x \in \Re^n, g_i \in \Re^n$$

$$\dot{x} = g_{S_2,1}u^1 + g_{S_2,2}u^2 + ... + g_{S_2,p}u^p, x \in \Re^n, g_i \in \Re^n$$

The number of inputs on the strata may differ since there could be additional constraints, other than the ones on M, which reduce the number of inputs to the system on that particular stratum. The b_i's are now chosen so that they span the tangent space of the bottom stratum. Recall that for a legged robot the bottom stratum corresponds to all the legs being in contact with the ground. The method for smooth systems can be applied to stratified systems only if the system is allowed to evolve on a common manifold. The bottom stratum is chosen as this manifold and the vector fields from the higher strata are assumed to be defined on the bottom stratum. An extended system is thus formed by using vector fields from all the strata. The extended system, however, does not represent the equations of motion on the bottom stratum since the vector fields from higher strata may not necessarily be part of the equations of motion for the system on the bottom stratum.

5.5.1 Condition for projecting vector fields onto the bottom stratum

The vector fields from the higher strata cannot always be directly used to describe the extended system on the bottom stratum. They need to satisfy certain conditions. Consider an example of a four-legged robot again. Let S_{123} and S_{1234} denote the stratum corresponding to feet 1,2,3 in contact and all feet in contact respectively. Consider a flow sequence of the form.

$$\begin{array}{ccc} e^{\alpha g_{l,4}} & e^{\beta g_{S_{123},1}} & e^{-\alpha g_{S_{l,4}}} \\ S_{1234} \rightarrow S_{123} & \text{on } S_{123} & S_{123} \rightarrow S_{1234} \end{array}$$

This sequence of flows can be visualized as a flow along $g_{l,4}$ which lifts up one leg, a flow along $g_{S_{123},1}$ and a flow along - $g_{l,4}$ which should bring the leg back into contact with the ground. If at the end of this flow, leg 4 is back in contact with the ground then the resultant flow can be described as a flow in S_{1234}. The above flow will not, however, bring the leg back into contact with the ground all the time. Let the flow along $g_{l,4}$ be a motion that changes only the height of the leg 4 above the ground and brings it into and out of contact with the ground. Then the flow $e^{\alpha g_{l,4}}$ will change the height of leg 4 from zero to say z_1. If the flow along $g_{S_{123},1}$ changes the height of the leg above the ground as well, then the flow $e^{-\alpha g_{l,4}}$ will not bring the leg back into contact with the ground. This can be avoided if the flow along any other strata is tangent to the bottom stratum. This condition can be represented more formally.

The bottom stratum is defined by a set of constraints $\Phi(x_1, x_2, ..., x_n) = \text{constant}$. A vector field *g* defined on a higher stratum can be projected down into the bottom stratum if

$$d\Phi.g = 0$$

This is equivalent to $g \in \Omega^{\perp}$ where $\Omega = span\{d\Phi_1, d\Phi_2, ..., d\Phi_k\}$ and *k* is the number of constraints defined for the bottom stratum.

If a vector field that needs to be projected down is not tangent to the bottom stratum a suitable transformation can be used to make it tangent to the bottom stratum. Let g_1 and g_2 be two vector fields such that $d\Phi.g_1 = f_1(x) \neq 0$ and $d\Phi.g_2 = f_2 \neq 0$. Then the vector field

$$\tilde{g}_1 = g_1 - \frac{f_1}{f_2} g_2$$

is tangent to the bottom stratum and can be used to define the extended system on the bottom stratum. We assume that the vector fields on the higher strata are tangent to the bottom strata or can be made tangent to the lower strata as above. The method for calculating the real inputs is the same now as for smooth systems. It differs only in the choice of vector fields which are now chosen from all the strata.

5.6 Gaits

The result of the method above is a sequence of inputs which may involve flows in different strata. Since different strata correspond to different feet in contact, frequent switching between strata may need to be carried out. The set of allowable switches between strata can be represented by a graph where the nodes represent the strata and the links indicate the ability to switch from one stratum to another. This switching is carried out by a set of independent inputs which raise or lower each of the legs. It is assumed that these inputs do not affect the rest of the equations of motion for the robot.

A graphical representation for a four legged robot stratification is shown in Figure 9. As expected, the transition between a stratum with 3 legs in contact and another with 3 legs in contact can only be achieved by going through the bottom stratum. This diagram does not represent the complete stratification for the four-legged robot since it does not include strata where less than 3 feet are in contact. These strata however correspond to unstable configurations for the robot since a four-legged kinematic robot standing on two legs would almost certainly fall down.

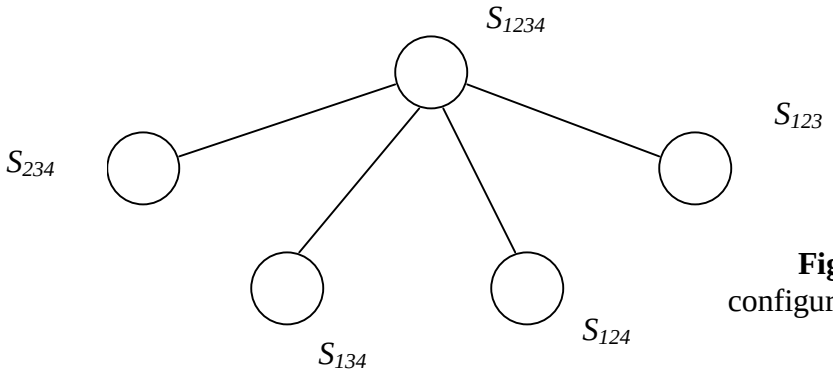


Figure 9: Stratified configuration space for a four-legged robot

The resultant flow along strata can sometimes be composed into a smaller sequence of flows. This is a consequence of the fact that if the Lie bracket of two vector fields is zero, then the flows corresponding to the two vector fields commute according to the Campbell-Baker-Hausdorff formula. Consider the following sequence of flows

$$e^{\alpha g_{l,4}} \quad e^{\beta g_{S_{123},1}} \quad e^{-\alpha g_{l,4}} \quad e^{\gamma g_{l,3}} \quad e^{\beta g_{S_{124},1}} \quad e^{-\gamma g_{l,3}}$$
$$S_{1234} \to S_{123} \quad \text{on } S_{123} \quad S_{123} \to S_{1234} \quad S_{1234} \to S_{124} \quad \text{on } S_{124} \quad S_{124} \to S_{1234}$$

$$e^{\alpha g_{l,4}} \quad e^{-\beta g_{S_{123},1}} \quad e^{-\alpha g_{l,4}} \quad e^{\gamma g_{l,3}} \quad e^{-\beta g_{S_{124},1}} \quad e^{-\gamma g_{l,3}}$$
$$S_{1234} \to S_{123} \quad \text{on } S_{123} \quad S_{123} \to S_{1234} \quad S_{1234} \to S_{124} \quad \text{on } S_{124} \quad S_{124} \to S_{1234}$$
$$[g_{S_{123},1}, g_{l,4}] = 0$$

If $[g_{S_{123},1}, g_{l,4}] = 0$ then $e^{\alpha g_{l,4}} e^{\beta g_{S_{123},1}} e^{-\alpha g_{l,4}} = e^{\alpha g_{l,4}} e^{-\alpha g_{l,4}} e^{\beta g_{S_{123},1}} = e^{\beta g_{S_{123},1}}$. The other flows can also be rearranged to get a resultant flow $e^{\beta g_{S_{123},1}} e^{\beta g_{S_{124},1}} e^{-\beta g_{S_{123},1}} e^{-\beta g_{S_{124},1}}$ in S_{1234} that is equivalent to a flow along the Lie bracket direction of two vector fields defining the equations of motion in two different strata S_{123} and S_{124}.

6 Examples

Two systems are now considered for which the equations of motion are formulated and inputs derived for a few desired motions. The two systems are a four-legged robot and a hybrid robot with two wheels and two legs. The approach here differs from [9] in that the allowable body velocities are used as the inputs to the system. The two systems are now discussed in greater detail.

6.1 A four-legged robot (quadruped)

The robot considered here is a four-legged kinematic robot. A few assumptions about the motion of the robot are made to simplify the equations of motion.

1. It is assumed that the center of mass of the robot stays at a constant height above the ground. This means that there is no motion of the center of mass in the z direction.
2. The only rotation allowed for the robot is about the z axis. Thus, the robot is not allowed to roll or pitch.
3. When the robot has only three feet in contact, the equations of motion are assumed to remain the same on that stratum regardless of the height of the free leg in the air. Thus, the only discontinuities in the equations of motion occur due to the transition between strata. The equations of motion are smooth on the same stratum.

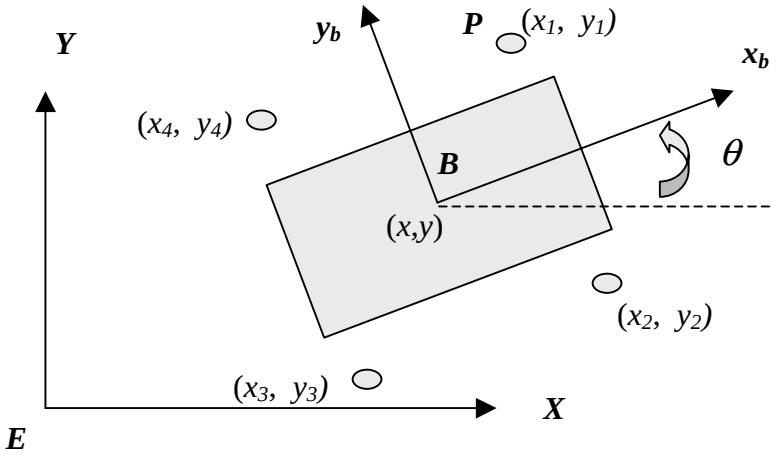


Figure 10: Notation for a four legged robot

6.1.1 Equations of motion

The equations of motion for the robot are formulated in terms of the contact points of the legs on the ground. A schematic representation for the robot is shown in Figure 10. The coordinates of the contact points for the legs (x_i, y_i), i = 1,2,3,4,are represented in the body frame ***B***. The position of the center of mass of the robot (x,y) and the orientation (θ) of the robot are represented in the inertial coordinate frame ***E***.

The two body velocities, viz., the velocity of the robot along $\boldsymbol{x_b}$ and the velocity of the robot along $\boldsymbol{y_b}$, and the angular rotation of the body $\dot{\theta}$ are chosen as the inputs to the system. On the bottom stratum there are 4 more inputs which allow the legs to come into and out of contact with the ground. These are independent inputs which do not influence the equations of motion and hence are not included in them. Since the velocity of the point of contact of the leg with the ground in the frame ***E*** is zero, the relative velocity of the point of contact with respect to the body fixed frame can be found in terms of the body velocities and angular velocity.

Let v_1^b represent the velocity of point of contact (P) of leg 1 in the body fixed frame. If leg 1 is on the ground, then the velocity of point P in the frame ***E*** is given by

$$v_1^e = Rv_c^b + Rv_1^b + R(\dot{\theta} \times r_p)$$

Here,

R is the 3x3 rotation matrix relating ***E*** and ***B***,

$v_c^b = \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \end{bmatrix}$ velocity of center of mass of robot in ***B*** (the body velocity),

v_1^b = relative velocity of point of contact in ***B***,

r_p = position of point of contact of leg 1 in ***B***,

But $v_1^e = 0$. So, $v_1^b = -v_c^b - (\dot{\theta} \times r_p)$

which gives the equations of motion for the points of contact of the legs in terms of the body velocities and the angular velocity of the robot. Using these equations for each leg and putting all the terms together, the equation 11 gives the equations of motion on the bottom stratum. Here u_1 and u_2 are the body velocities along $\boldsymbol{x_b}$ and $\mathbf{y_b}$ respectively while u_3 is the angular velocity of the robot ($\dot{\theta}$).

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{x}_1 \\ \dot{y}_1 \\ \dot{x}_2 \\ \dot{y}_2 \\ \dot{x}_3 \\ \dot{y}_3 \\ \dot{x}_4 \\ \dot{y}_4 \end{pmatrix} = \begin{bmatrix} c\theta & -s\theta & 0 \\ s\theta & c\theta & 0 \\ 0 & 0 & 1 \\ -1 & 0 & y_1 \\ 0 & -1 & -x_1 \\ -1 & 0 & y_2 \\ 0 & -1 & -x_2 \\ -1 & 0 & y_3 \\ 0 & -1 & -x_3 \\ -1 & 0 & y_4 \\ 0 & -1 & -x_4 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \qquad (11)$$

On a stratum with leg 1 not in contact, the equations of motion are given as

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{x}_1 \\ \dot{y}_1 \\ \dot{x}_2 \\ \dot{y}_2 \\ \dot{x}_3 \\ \dot{y}_3 \\ \dot{x}_4 \\ \dot{y}_4 \end{pmatrix} = \begin{bmatrix} c\theta & -s\theta & 0 & 0 & 0 \\ s\theta & c\theta & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ -1 & 0 & y_2 & 0 & 0 \\ 0 & -1 & -x_2 & 0 & 0 \\ -1 & 0 & y_3 & 0 & 0 \\ 0 & -1 & -x_3 & 0 & 0 \\ -1 & 0 & y_4 & 0 & 0 \\ 0 & -1 & -x_4 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix} \qquad (12)$$

where u_4 and u_5 are two independent inputs on S_{234} which control the position of the free leg in the air. Each stratum with one of the legs not in contact can be described by a similar set of equations.

6.1.2 Choice of basis

For the basis which spans the tangent space of the bottom stratum, 9 vector fields are needed. These can be chosen from the vector fields in the equations of motion for all the strata. The vector fields chosen here include the first three from the equations of motion on the bottom stratum and the last two from the equations of motion for each of the strata.

Thus, the set of vector fields used to form the extended system are

$$g_1 = \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \\ -1 \\ 0 \\ -1 \\ 0 \\ -1 \\ 0 \\ -1 \\ 0 \end{bmatrix}, g_2 = \begin{bmatrix} -\sin\theta \\ \cos\theta \\ 0 \\ 0 \\ -1 \\ 0 \\ -1 \\ 0 \\ -1 \\ 0 \\ -1 \end{bmatrix}, g_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ y_1 \\ -x_1 \\ y_2 \\ -x_2 \\ y_3 \\ -x_3 \\ y_4 \\ -x_4 \end{bmatrix}, g_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, g_5 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (13)$$

$$g_6 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, g_7 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, g_8 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, g_9 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, g_{10} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, g_{11} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad (14)$$

The extended system can be formed as $\dot{x} = b_1 v^1 + b_2 v^2 + ... + b_{11} v^{11}$ where $b_i = g_i$.
So, the extended system on the bottom stratum is given as

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{x}_1 \\ \dot{y}_1 \\ \dot{x}_2 \\ \dot{y}_2 \\ \dot{x}_3 \\ \dot{y}_3 \\ \dot{x}_4 \\ \dot{y}_4 \end{pmatrix} = \begin{bmatrix} c\theta & -s\theta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ s\theta & c\theta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & y_1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & -x_1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & y_2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & -x_2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & y_3 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & -x_3 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ -1 & 0 & y_4 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & -x_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \\ v_8 \\ v_9 \\ v_{10} \\ v_{11} \end{bmatrix}$$

Given an initial and desired position for the system, the set of inputs required to steer the system can now be found. The complete method is illustrated for one example and simulation results for other examples are presented.

6.1.3 Translation

Suppose the robot has to move from $x_i = (0,0,0,1,1,1,-1,-1.1,-1,-1.1,1)$ to $x_d = (0.1,0.1,0,1,1,1,-1,-1.1,-1,-1.1,1)$, i.e. it has to move along a diagonal path. We define a path that goes from x_i to x_d as $\gamma(t) = (0.1t,0.1t,0,1,1,1,-1,-1,-1,-1,1)$. The fictitious inputs can now be solved for by solving the set of simultaneous equations given by

$$\dot{\gamma}(\mathrm{t}) = b_1(\gamma(t))\, v^1 + ... + b_9(\gamma(t))\, v^9$$

The set of fictitious inputs are then given as

$$\begin{aligned} v_j &= 0.1,\ j = 1,2,4,...,11 \\ v_3 &= 0 \end{aligned} \qquad (14)$$

The next step is to determine the manner in which the Philip Hall coordinates evolve. The differential equations which describe this evolution need to be formulated. Now

$$Ad_{e^{-h_1 b_1} e^{-h_2 b_2}} b_3 \dot{h}_3(\mathrm{t}) = Ad_{e^{-h_1 b_1}} Ad_{e^{-h_2 b_2}} b_3 \dot{h}_3(\mathrm{t})$$

$$Ad_{e^{-h_1 b_1}} Ad_{e^{-h_2 b_2}} b_3 \dot{h}_3(\mathrm{t}) = Ad_{e^{-h_1 b_1}}((b_3 - h_2[b_2, b_3])\dot{h}_3) = (b_3 - h_2[b_2, b_3] - h_1[b_1, b_3] + h_1 h_2[b_1, [b_2, b_3]])\dot{h}_3 \qquad (15)$$

However since we assume the system is nilpotent of order 2, the last term in equation (15) which is a third order bracket drops out. Therefore,

$$Ad_{e^{-h_1 b_1}} Ad_{e^{-h_2 b_2}} b_3 \dot{h}_3 \text{ (t)} = Ad_{e^{-h_1 b_1}} ((b_3 - h_2[b_2, b_3])\dot{h}_3) \approx (b_3 - h_2[b_2, b_3] - h_1[b_1, b_3])\dot{h}_3$$

On calculating the Lie brackets,

$$Ad_{e^{-h_1 b_1}} Ad_{e^{-h_2 b_2}} b_3 \dot{h}_3 \text{ (t)} = Ad_{e^{-h_1 b_1}} ((b_3 - h_2[b_2, b_3])\dot{h}_3) \approx (b_3 - h_2 b_1 + h_1 b_2)\dot{h}_3$$

The other expressions can be similarly computed to get, for j = 4,6,8,10

$$Ad_{e^{-h_1 b_1} \ldots\ldots e^{-h_{j-1} b_{j-1}}} b_j \dot{h}_j \ (t) = (b_j - h_3 b_{j+1})\dot{h}_j$$

and for j=5,7,9,11

$$Ad_{e^{-h_1 b_1} \ldots\ldots e^{-h_{j-1} b_{j-1}}} b_j \dot{h}_j \ (t) = (b_j + h_3 b_{j-1})\dot{h}_j$$

Thus, by equating the coefficients of b_j to v_j we have a set of differential equations for the Philip hall co-ordinates

$$\begin{aligned}
&\dot{h}_1 - h_2 \dot{h}_3 = v_1 \\
&\dot{h}_2 + h_1 \dot{h}_3 = v_2 \\
&\dot{h}_3 = v_3 \\
&\dot{h}_j + h_3 \dot{h}_{j+1} = v_j \ , \ j = 4,6,8,10 \\
&\dot{h}_j - h_3 \dot{h}_{j-1} = v_j \ , \ j = 5,7,9,11 \\
&h_j(0) = 0, \ j = 1,2,....,11
\end{aligned} \qquad (16)$$

Since v_3=0, equation (16) can be easily solved by a set of successive integrations to give

$$h_j(1) = 0.1, \ j = 1,2,4,5,6,....,11$$

$$h_3(1) = 0$$

This corresponds to a flow $e^{0.1b_{11}} e^{0.1b_{10}} .....e^{0.1b_4} e^{0.1b_2} e^{0.1b_1}$. An example explains the method of converting this flow to a sequence of real inputs. Consider a flow, e.g. $e^{0.1b_{11}}$,along the vector field b_{11}. This flow can be executed by setting the input corresponding to the vector b_{11} equal to 0.1 for a period of 1s when the robot is on the stratum for which b_{11} lies in the equations of motion of the robot. In terms of movement on different strata and the resultant gait, this flow can be interpreted as the following sequence of events

1. Pick up leg 4 and move it ahead in the x_b and y_b directions by 0.1.This motion occurs on stratum S_{234}.
2. Replace leg 4 on the ground,
3. Pick up legs 3, 2 and 1 and carry out steps 1 and 2 for each of them in turn,
4. Move the body forward by 0.1 in the *x* and *y* direction with all feet in contact with the ground.

It should be noted that the order in which the legs are to be picked up and replaced is not fixed. This is because the flows along the corresponding vector fields commute and hence can be executed in any order. Implicit in the sequence given above is the switching that needs to be done to move from one stratum to another. As mentioned earlier, this is achieved by a set of independent inputs which pick up and replace the legs in contact with the ground as required.

Interesting note

The motion along the x and y directions for the free leg can be combined into a single flow. This is possible because the vector fields corresponding to these two flows commute, i.e. their Lie bracket is zero and the two vector fields are on the same stratum. Also, note that the motion of the body in the x and y directions can be executed simultaneously as well in which case the robot moves directly towards the desired position in a straight line instead of executing two separate motions in the x and y directions.

The simulation results for this example are presented in Figure 11 below. Figure 15c is a plot of the path of the center of mass of the robot and Figure 15b is a plot of the position coordinates of one of the contact points vs. time. Since, the position coordinates of the other legs vary in a similar fashion their plots are not presented here. Also included is a plot of the x position coordinates for all the legs vs. time in Figure 15a. This gives an idea about the relative phasing of the motion of the legs.

For the robot to execute a larger step, say from $x_i = (0,0,0,1,1,1,-1,-1.1,-1,-1.1,1)$ to $x_d = (1,1,0,1,1,1,-1,-1.1,-1,-1.1,1)$, the motion is carried out in 10 incremental steps. There are two reasons to carry out the motion in smaller steps.

1. Smaller steps lead to a more stable robot. Further, it may be physically impossible for an actual robot to achieve a motion of 1 unit along any direction in a single step.
2. Because of the approximation that the system is nilpotent of order 2, the errors are much larger for a bigger step than for a smaller step.

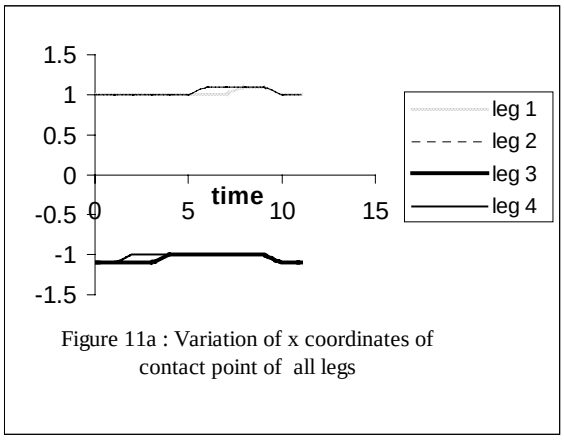


Figure 11a : Variation of x coordinates of contact point of all legs

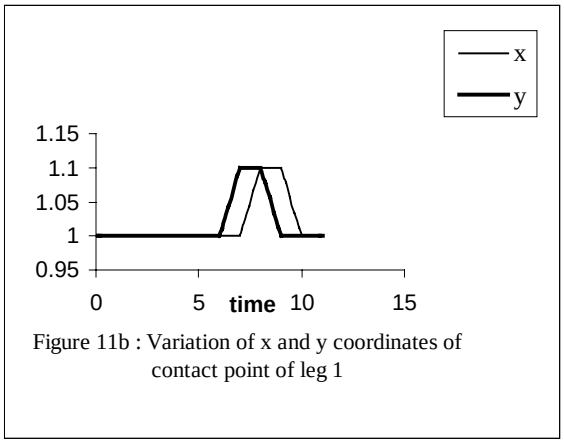


Figure 11b : Variation of x and y coordinates of contact point of leg 1

Figure 11: Pure translation of the robot.

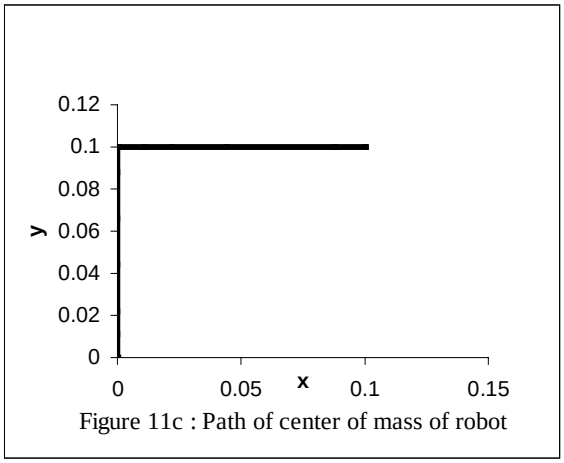


Figure 11c : Path of center of mass of robot

Figure 11: Pure translation of the robot.

6.1.4 Pure Rotation

A pure rotation of the robot is carried out for the second simulation. Here, the objective is to move from $x_i = (0,0,0,1,1,1,-1,-1.1,-1,-1.1,1)$ to $x_d = (0,0,0.1,1,1,1,-1,-1.1,-1,-1.1,1)$.
We define a path that goes from x_i to x_d as $\gamma\ (t) = (0,0,0.1t,1,1,1,-1,-1,-1,-1,1)$.
The differential equations describing the evolution of the Phillip Hall coordinates remain the same although the fictitious inputs now differ. The new fictitious inputs are

$$v_1, v_2 = 0, v_3 = 0.1, v_4 = -0.1, v_5 = 0.1,$$
$$v_6 = 0.1, v_7 = 0.1, v_8 = 0.1, v_9 = -0.11,$$
$$v_{10} = -0.1, v_{11} = -0.11$$

Using these and an approximation that $(1+h_3^2 \approx 1)$ which simplifies the differential equations, the Philip Hall coordinates are found to be

$$h_1(1) = 0, h_2(1) = 0, h_3(1) = 0.1, h_4(1) = -0.105,$$
$$h_5(1) = 0.095, h_6(1) = 0.095, h_7(1) = 0.105, h_8(1) = 0.105,$$
$$h_9(1) = -0.095, h_{10}(1) = -0.095, h_{11}(1) = -0.105$$

This corresponds to a flow $e^{h_{11}b_{11}} e^{h_{10}b_{10}} .....e^{h_3 b_3}$. The simulation results for this flow are shown in Figure 12. Figure 12a plots the variation of the orientation of the robot with time while the other figures are plots of the positions of all the legs. Again if the robot needs to move from $x_i = (0,0,0,1,1,1,-1,-1,-1,-1,1)$ to $x_d = (0,0,1,1,1,1,-1,-1,-1,-1,1)$, the motion is carried out in steps of 0.1^c each time.

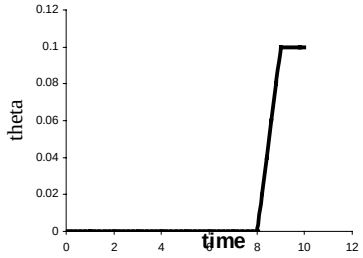


Figure 12a : Variation of theta

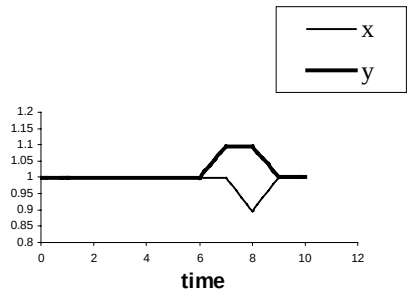


Figure 12 b : Variation of x and y coordinates of contact point of leg 1

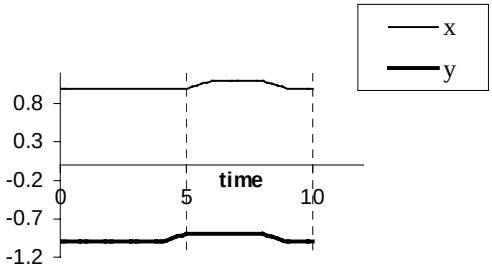


Figure 12c : Variation of x and y coordinates of contact point of leg 2

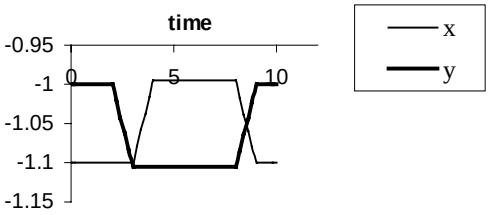


Figure 12d : Variation of x and y coordinates of contact point of leg 3

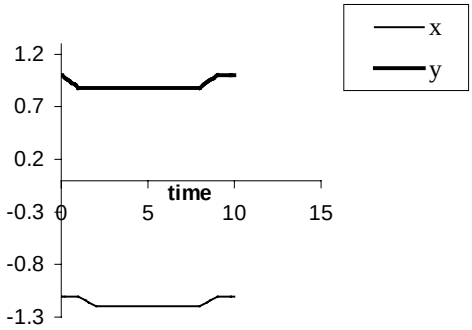


Figure 12e : Variation of x and y coordinates of contact point of leg 4

Figure 12: Pure Rotation of the robot

6.1.5 Rotation and translation

The problem of simultaneous rotation and translation can also be solved using this approach. The results for going from $x_i = (0,0,0,1,1,1,-1,-1,-1,-1,1)$ to $x_d = (0.1,0.1,0.1,1,1,1,-1,-1,-1,-1,1)$ are shown in Figure 13. Figures 13a and 13b show the path of the center of mass of the robot and the orientation of the robot respectively while Figures 13c to 13f plot the motion of the legs. There is an error of about 5% in the

x and y coordinates of the robot's final position. This error is caused because in solving for the fictitious inputs, the value of θ is assumed to remain constant for the incremental motion while it actually changes through the motion. The small error can be reduced by a second iteration of the algorithm from the new position to the original desired position. Another approach to solving this problem would be to decompose it into two separate problems, i.e., a translation and then a rotation from the new position that is reached after the translation. The advantage in using this approach is that the value of θ used for calculating the fictitious inputs during translation would be the same as the actual value of θ during translation. The two flows could be rearranged to get a resultant flow which would reach the final desired position in a single step. In this case, the sequence of inputs required is just a concatenation of the two sequences of inputs derived for the translation and rotation respectively. The method can then be iterated from the new position to generate a new sequence of inputs which moves it to the new desired position. Since the value of θ has now changed the new inputs would be different for each iteration.

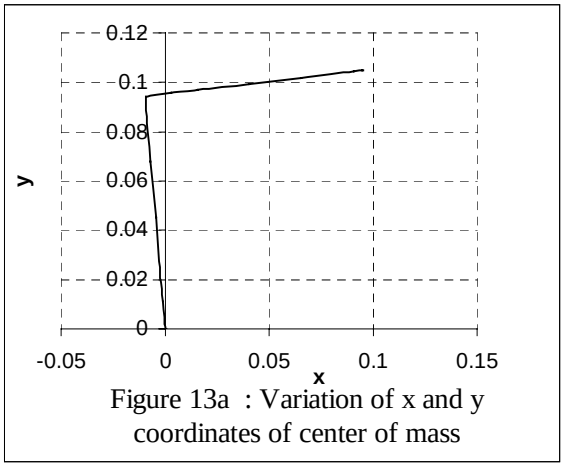


Figure 13a : Variation of x and y coordinates of center of mass

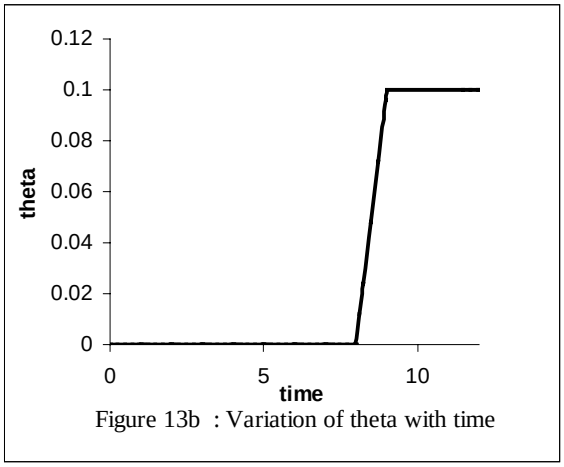


Figure 13b : Variation of theta with time

Figure 13: Rotation and translation of the robot

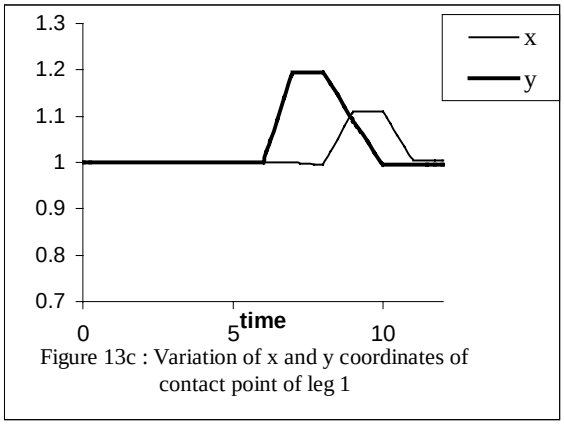


Figure 13c : Variation of x and y coordinates of contact point of leg 1

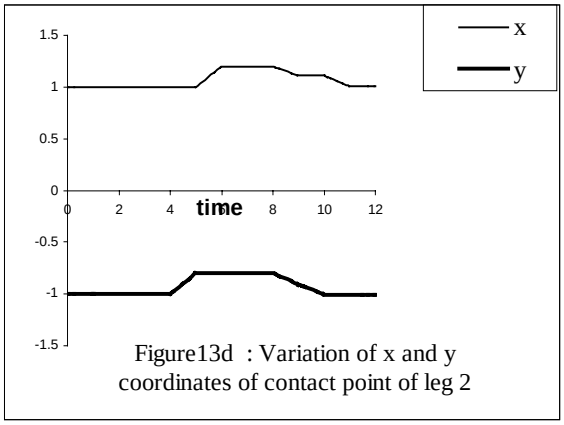


Figure13d : Variation of x and y coordinates of contact point of leg 2

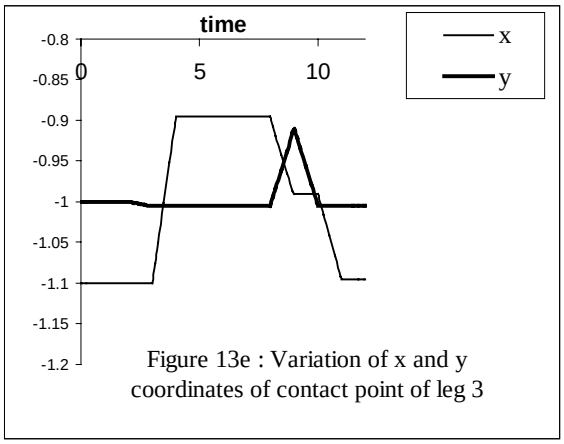


Figure 13e : Variation of x and y coordinates of contact point of leg 3

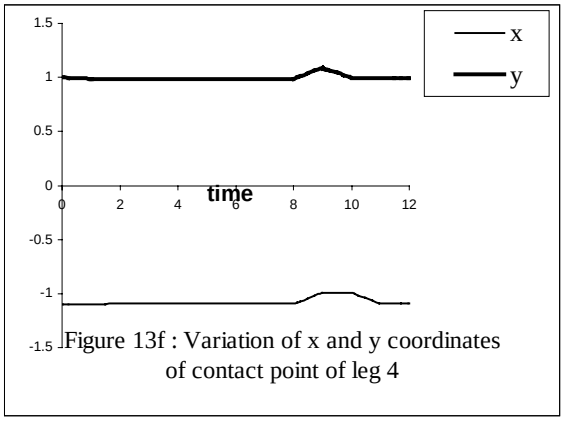


Figure 13f : Variation of x and y coordinates of contact point of leg 4

Figure 13: Rotation and translation of the robot

6.2 Wheel-Legged Hybrid Robot

Consider a robot with two wheels and two legs. The robot is shown in Figure 14. The two wheels are placed on the same axle. The axle can be turned though a steering angle ϕ. The robot now has a set of allowable velocities and hence the body velocities cannot be used as inputs. The allowable velocities arise from the constraint that the velocity of the center of the axle must always be perpendicular to the axle. The three inputs chosen for the robot are the speed (V)of the center of the axle called the drive speed, rate of change of the steering angle ($\dot{\phi}$) and the angular velocity of the robot (ω). There are now 3 strata corresponding to one or both of the feet remaining in contact. Both the feet are not allowed to be off the ground at the same time. The configuration space for the robot includes the position of the center of mass, the orientation of the robot, the positions of the contact points of each of the feet and the steering angle.

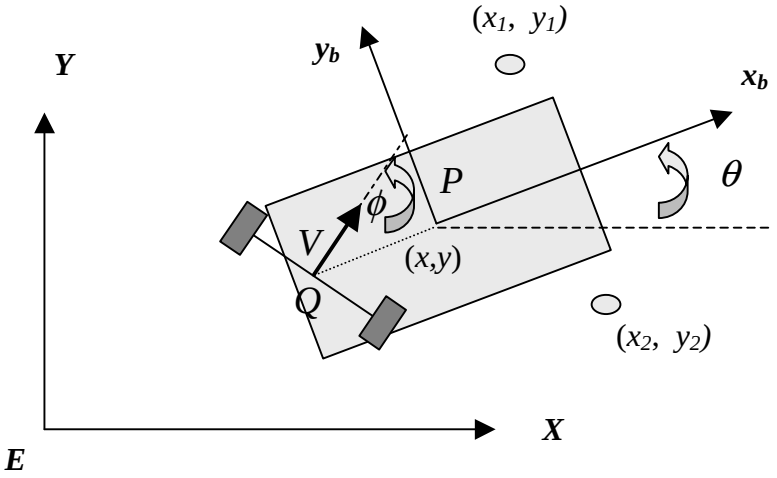


Figure 14: A wheel-legged mobile robot

6.2.1 Equations of motion

To write the equations of motion for this robot, the concepts presented in section 5.2 are used.

$$\text{Here } \xi = \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \\ \dot{\theta} \end{bmatrix} = f(x)u^w = \begin{bmatrix} \cos(\phi) & 0 \\ \sin(\phi) & l \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V \\ \omega \end{bmatrix} \qquad (17)$$

$$\text{Also, } \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \\ \dot{\theta} \end{bmatrix} = A(x)\xi \qquad (18)$$

From equations (17) and (18),

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos(\theta+\phi) & -l\sin\theta \\ \sin(\theta+\phi) & l\cos\theta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V \\ \omega \end{bmatrix} \qquad (19)$$

Also, we have the equations for the shape inputs for the contact point of a leg in contact with the ground in terms of the body velocities as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{y}_1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & y_1 \\ 0 & -1 & -x_1 \end{bmatrix} \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \\ \dot{\theta} \end{bmatrix}$$

Using equation (17) and a transformation to express the velocities in the body fixed frame of reference we get,

$$\begin{bmatrix} \dot{x}_1 \\ \dot{y}_1 \end{bmatrix} = \begin{bmatrix} -\cos\phi & y_1 \\ -\sin\phi & -x_1 \end{bmatrix} \begin{bmatrix} V \\ \omega \end{bmatrix} \qquad (20)$$

For the wheels, the shape input is the steering angle which is an independent input. Putting together equations (19) and (20) and the steering input,t he equations of motion for the robot in the bottom stratum are

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{x}_1 \\ \dot{y}_1 \\ \dot{x}_2 \\ \dot{y}_2 \\ \dot{\phi} \end{pmatrix} = \begin{bmatrix} \cos(\theta+\phi) & -l\sin\theta & 0 \\ \sin(\theta+\phi) & l\cos\theta & 0 \\ 0 & 1 & 0 \\ -\cos\phi & y_1 & 0 \\ -\sin\phi & -x_1 & 0 \\ -\cos\phi & y_2 & 0 \\ -\sin\phi & -x_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} V \\ \omega \\ \dot{\phi} \end{bmatrix}$$

where the notation for the legs is the same as that for the four-legged robot case. Here, *l* is the distance from point P to point Q and is taken as 1 and $\omega = \dot{\theta}$.

The equations of motion with leg 1 in the air are

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{x}_1 \\ \dot{y}_1 \\ \dot{x}_2 \\ \dot{y}_2 \\ \dot{\phi} \end{pmatrix} = \begin{bmatrix} \cos(\theta+\phi) & -l\sin\theta & 0 & 0 & 0 \\ \sin(\theta+\phi) & l\cos\theta & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ -\cos\phi & y_2 & 0 & 0 & 0 \\ -\sin\phi & -x_2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} V \\ \omega \\ \dot{\phi} \\ u_4 \\ u_5 \end{bmatrix}$$

A similar set of equations can be written for the stratum where leg 2 is not in contact with the ground. The first 7 vector fields required to span the tangent space of the bottom stratum are

$$g_1 = \begin{bmatrix} \cos(\theta+\phi) \\ \sin(\theta+\phi) \\ 0 \\ -\cos\phi \\ -\sin\phi \\ -\cos\phi \\ -\sin\phi \\ 0 \end{bmatrix}, g_2 = \begin{bmatrix} -l\sin(\theta) \\ l\cos(\theta) \\ 1 \\ y_1 \\ -x_1 \\ y_2 \\ -x_2 \\ 0 \end{bmatrix}, g_3 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, g_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, g_5 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, g_6 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, g_7 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

The eighth vector field is chosen as $[g_1, g_3]$. Thus, $g_8 = \begin{bmatrix} \sin(\theta+\phi) \\ -\cos(\theta+\phi) \\ 0 \\ -\sin\phi \\ \cos\phi \\ -\sin\phi \\ \cos\phi \\ 0 \end{bmatrix}$,

The complete extended system on the bottom stratum is
$\dot{x} = b_1 v^1 + b_2 v^2 + ... + b_8 v^8, b_i = g_i$. The system differs from the previous system in that a Lie bracket direction is required to span the tangent space for the bottom stratum.

6.2.2 Translation along y-axis

For a robot starting from $x_i = (0,0,0,1,1,1,-1,0)$ and moving to $x_d = (0,0.1,0,1,1,1,-1,0)$, the direction in which it needs to move is a Lie-Bracket direction which does not correspond to any of the given inputs. We define a path that goes from x_i to x_d as
$\gamma(t) = (0,0.1t,0,1,1,1,-1,0)$. The calculation of the required inputs is now illustrated for such a motion. Using the extended system the fictitious inputs are

$$v_1 = 0, v_2 = 0, v_3 = 0, v_4 = 0,$$
$$v_5 = 0.1, v_6 = 0, v_7 = 0.1, v_8 = -0.1$$

The corresponding differential equations for the Philip Hall coordinates are

$$\dot{h}_1 - h_3\dot{h}_8 - h_2\dot{h}_8 = v_1$$
$$\dot{h}_2 = v_2$$
$$\dot{h}_3 = v_3,$$
$$\dot{h}_j + h_2\dot{h}_{j+1} = v_j, \; j = 4,6$$
$$\dot{h}_j - h_2\dot{h}_{j-1} = v_j, \; j = 5,7$$
$$\dot{h}_8 - h_1\dot{h}_3 - h_1\dot{h}_2 = v_8$$

Putting in the fictitious coordinates and solving for the Philip Hall coordinates,

$$h_1(1) = 0, h_2(1) = 0, h_3(1) = 0, h_4(1) = 0,$$
$$h_5(1) = 0.1, h_6(1) = 0, h_7(1) = 0.1, h_8(1) = -0.1$$

The system needs to move in the direction of the vector field $-b_8 = -[b_1, b_3] = [b_3, b_1]$, which is a Lie bracket direction, with a magnitude of 0.1. This corresponds to a sequence of inputs

$$u_3 = \sqrt{0.1} \text{ for t} = 0 \text{ to } 1\,\text{s}$$
$$u_1 = \sqrt{0.1} \text{ for t} = 1 \text{ to } 2\,\text{s}$$
$$u_3 = -\sqrt{0.1} \text{ for t} = 2 \text{ to } 3\,\text{s}$$
$$u_1 = -\sqrt{0.1} \text{ for t} = 3 \text{ to } 4\,\text{s}$$

Using the Campbell-Baker-Hausdorff formula for flows it can be proved that this sequence of inputs actually gives rise to a flow

$$e^{h_3[g_3,g_1]}e^{\alpha[g_3,[g_3,g_1]]}e^{\alpha[g_1,[g_3,g_1]]} = e^{h_3[g_3,g_1]}e^{\alpha[g_3,[g_3,g_1]]} = e^{h_3[g_3,g_1]}e^{-\alpha g_1}$$

where $\alpha = \frac{1}{2}h_3^{\frac{3}{2}}$

Thus an additional input of $u_1 = \frac{1}{2}(0.1)^{\frac{3}{2}}$ for t = 4 to 5 s is required to give the required total flow $e^{h_3[g_3,g_1]}$. This input can be added to the previous input since they represent motion along the same vector field to get a single input of

$$u_1 = -\sqrt{0.1} + \frac{1}{2}(0.1)^{\frac{3}{2}} \text{ for t} = 3 \text{ to } 4\,\text{s}.$$

In addition, two more flows are required along b_5 and b_7 to give the total flow for the motion $e^{h_8 b_8}e^{h_7 b_7}e^{h_5 b_5}$. The simulation results for the system are shown in Figure 15. Figure 15a shows the path of the robot. Figure 15b and 15c plots the motion of the legs. Figure 15d plots the variation of the steering angle ϕ. It can be seen that the steering angle changes to allow the robot to move in the Lie bracket direction. Figure 15e plots the drive input V which moves the robot in a direction at an angle of ϕ to the x axis. This results in the desired motion in the y direction but also leads to a motion in the x direction. The steering angle then changes back to zero and the robot moves backward to correct for the change in the x-coordinate. This type of motion is similar to the parallel parking of a car.

In terms of a gait representation, the sequence of flow can be visualized as follows

1. Change the steering angle to $\phi = \sqrt{0.1}^{c}$ – this motion occurs on bottom stratum
2. Move along vector field g_1 which corresponds to moving in a direction that is at an angle ϕ with respect to the x-axis of the inertial frame.
3. Change the steering angle back to zero
4. Move in negative x direction
5. Pick up and reset leg 2 to its initial position and then replace it on the ground
6. Repeat step 5 for leg1.

The order in which steps 5 and 6 are executed can be interchanged since the flows corresponding to these two motions commute.

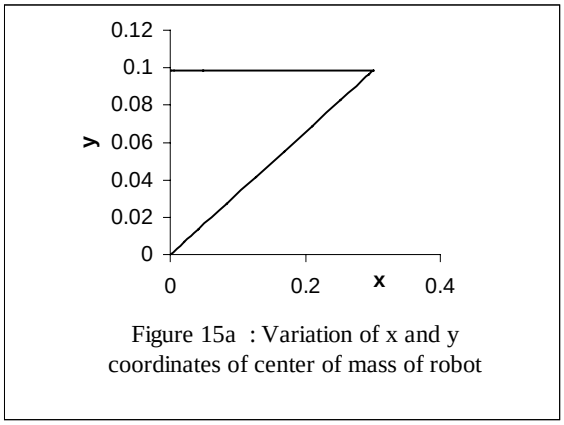


Figure 15a : Variation of x and y coordinates of center of mass of robot

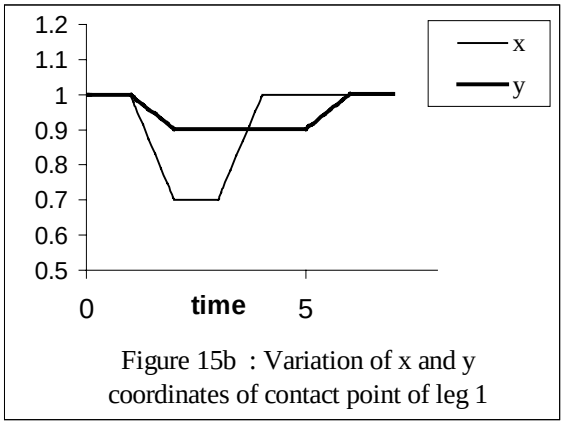


Figure 15b : Variation of x and y coordinates of contact point of leg 1

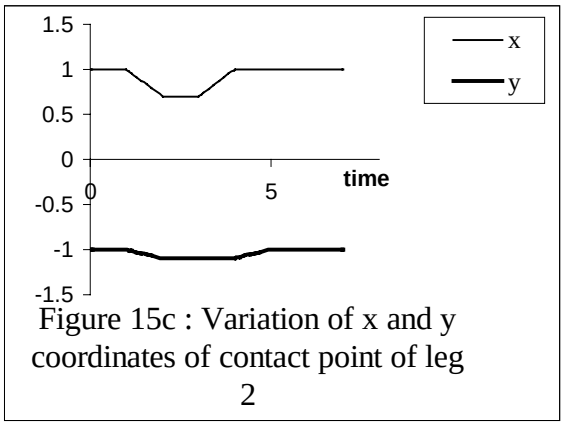


Figure 15c : Variation of x and y coordinates of contact point of leg 2

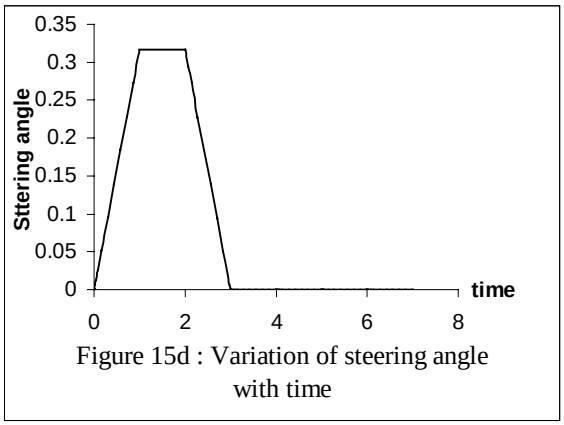


Figure 15d : Variation of steering angle with time

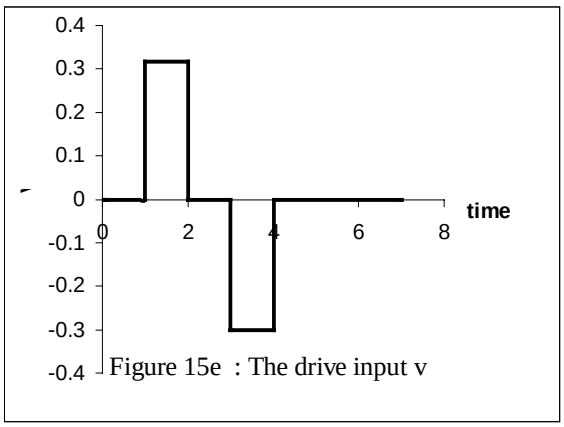


Figure 15e : The drive input v

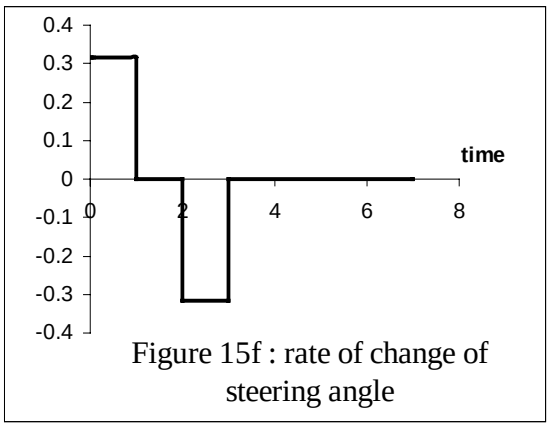


Figure 15f : rate of change of steering angle

Figure 15: Translation of a hybrid robot along the y-axis

6.2.3 Pure rotation

For the robot to move from $x_i = (0,0,0,1,1,1,-1,0)$ to $x_d = (0,0,0.1,1,1,1,-1,0)$, the required fictitious inputs are

$$v_1 = 0, v_2 = 0.1, v_3 = 0, v_4 = -0.1,$$
$$v_5 = 0, v_6 = 0.1, v_7 = 0, v_8 = 0.1$$

and the Philip Hall coordinates after solving the appropriate differential equations are given as

$$h_1(1) = 0.005, h_2(1) = 0.1, h_3(1) = 0, h_4(1) = -0.1,$$
$$h_5(1) = -0.005, h_6(1) = 0.1, h_7(1) = 0.005, h_8(1) = 0.1$$

The resultant flow is of the form $e^{h_8 b_8} e^{h_7 b_7} e^{h_6 b_6} e^{h_5 b_5} e^{h_4 b_4} e^{h_3 b_3} e^{h_2 b_2} e^{h_1 b_1}$. The simulation results for the system are shown in Figure 16. The flow involves a motion along the Lie bracket direction and this must be executed by a sequence of inputs similar to the ones given above. In terms of a gait representation, the sequence of flow can be visualized as follows

1. Move along vector field g_1 which corresponds to moving in a direction that is at an angle ϕ with respect to the x-axis of the inertial frame.
2. Change the steering angle to $\phi = \sqrt{0.1}^c$ – this motion occurs on bottom stratum
3. Move in negative *x* direction
4. Change the steering angle back to zero
5. Pick up and reset leg 2 to its initial position and then replace it on the ground
6. Repeat step 5 for leg1
7. Rotate the body through the required angle
8. Move along positive *x* direction

The order in which steps 5 and 6 are executed can be interchanged since the flows corresponding to these two motions commute. The resultant error in carry out this motion is less than 2 % in the x and y directions.

To execute a larger turn, the same series of turns can be executed each time. The fictitious inputs need to be recalculated each time since the orientation of the robot keeps on changing.

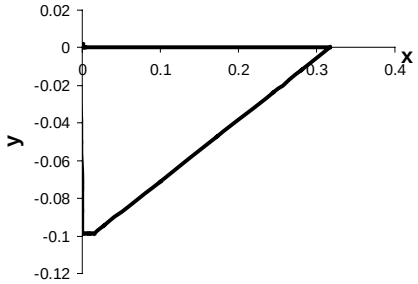


Figure 16a : Variation of x and y coordinates of center of mass of robot

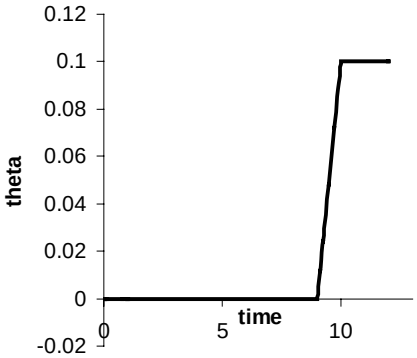


Figure 16b : Theta vs. time

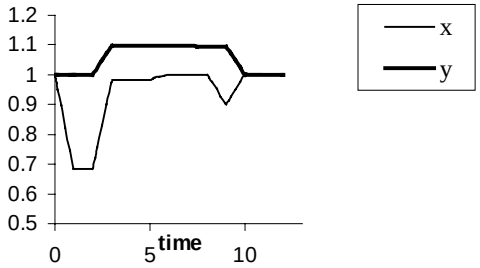


Figure 16c : Variation of x and y coordinates of contact point of leg 1

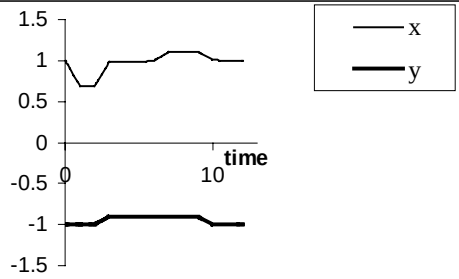


Figure 16d : Variation of x and y coordinates of contact point of leg 2

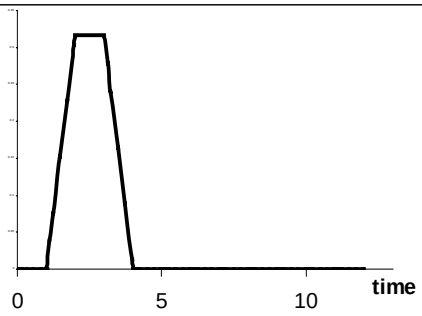


Figure 16e : Variation of steering angle vs. time

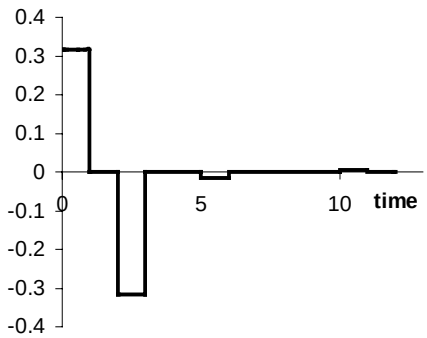


Figure 16f : Drive input V

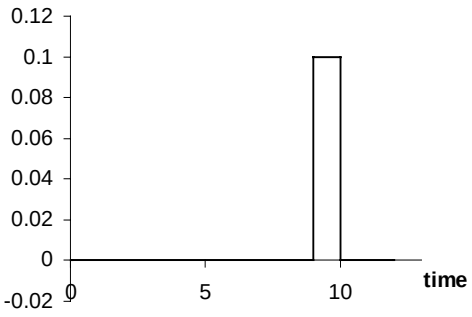


Figure 16g : Angular velocity of the robot

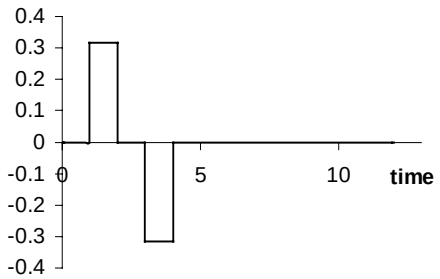


Figure 16h : Rate of change of steering angle (Input 3) vs. time

Figure 16: Rotation of a hybrid robot

7 Conclusion

In this report, a basic framework for studying locomotion of modular robots has been laid down. A method for enumerating distinct modular robot configurations has been presented and applied to locomotion. Once the distinct configurations are known, the equations of motion can be written down easily by combining the equation for different modules appropriately. A motion-planning algorithm for drift free systems can then be used to generate a set of inputs which steer the system to a desired state. The method can be applied to a combination of modules as well as was illustrated by its application to a wheel-legged hybrid robot. The accuracy of the method is limited when a nilpotent approximation is made. The accuracy of the algorithm can be improved by successive iteration to converge to the desired position. The algorithm can also be applied to follow a trajectory. The trajectory is broken up into a set of small motions and the appropriate inputs for each of these motions are generated.

The algorithm does not give rise to statically stable gaits automatically. The choice of a particular gait can be based on its stability which the algorithm does not guarantee. Further, the method cannot as yet be applied to systems with drift. The method needs to be extended to apply to dynamic systems as well.

References

[1] M. Yim, "A Reconfigurable Modular Robot with Many Modes of Locomotion", Proc. of the JSME Int. Conf. on Advanced Mechatronics, pp. 283-288 Tokyo, Japan 1993.

[2] K. Kotay, D. Rus, Task-reconfigurable robots for navigation and manipulation in International Conference on Intelligent Robots and Systems(IROS'97).

[3] K. Kotay, D. Rus, M. Vona, C. McGray, The Self-reconfiguring Robotic Molecule: Design and Control Algorithms in Algorithmic Foundations of Robotics, eds. P. Agrawal, L. Kavraki, M. Mason, A.K. Peters 1998

[4] G. S. Chirikjian, A.Pamecha and I. Ebert-Uphoff, "Evaluating efficiency of self-reconfiguration in a class of modular robots." Journal of Robotic Systems13, No. 5(May 1996), International Conference of Robotics and Automation (May 1996).

[5] S. Murata, H. Kurokawa, E. Yoshida, K. Tomita, S. Kokaji, A 3-D Self-Reconfigurable Structure, Proceedings of the 1998 IEEE International Conference on Robotics and Automation, Belgium.

[6] S. Murata, H. Kurokawa, S. Kokaji, Self-Assembling Machine, Proc. IEEE Conference on Robotics and Automation, 1994, 441-448.

[7] I.M. Chen, and J. Burdick, Enumerating the Non-Isomorphic Assembly Configurations of a Modular Robotic System, Int. J. of Robotics Research, vol. 17, no. 7, pp. 702-719.

[8] G. Lafferriere, H. Sussmann, Motion Palnning for Controllable Systems Without Drift, Proceedings of the IEEE International Conference on Robotics and Automation, Sacramento, California, April 1991.

[9] B. Goodwine, and J. Burdick, Trajectory Generation for Legged Robotic Systems, Proc. 1997 IEEE Int. Conf. on Robotics and Automation.