

Toward The Automatic Control of Robot Assembly Tasks via Potential Functions: The Case of 2-D Sphere Assemblies

Louis L. Whitcomb, Daniel E. Koditschek, João B. D. Cabrera*

Department of Electrical Engineering, Yale University

Abstract

A new approach to the problem of controlling automated assembly tasks using artificial potential functions is described. A simple class of tasks, 2-D sphere assemblies, is examined. A primitive constructive theory for the control of this class of tasks is presented. Preliminary computer simulations demonstrate the new approach may provide surprisingly good performance.

1 Introduction

This paper addresses the automatic generation of actuator commands for a robot manipulator that result in the motion of a collection of rigid-body parts from dis-assembled initial configurations to an assembled final configuration. There is a large literature in assembly that falls outside the domain of motion planning [2]. Within the motion planning literature, tradition divides the problem into an off-line geometric "trajectory-planning" phase and an on-line "trajectory-tracking" control phase, in which pre-planned paths are tracked by conventional robot controllers [7, 1]. Recent authors, e.g. [1], have begun to construct a formal connection between purely geometric "planning" algorithms and the underlying dynamics ("control") of the workpieces. In contrast, our approach to motion planning makes use of artificial potential functions [3, 8, 10, 9, 4] as a means of unifying the planning and control stages in the very statement of the problem: find a feedback law that brings the system to the desired final configuration from almost every initial state. We have shown in previous work that this is a completely general alternative approach in principle [6], and have reported our success in various constructive instances of these problems as well [11, 13]. This paper applies our potential field methods to the problem of multibody assembly.

Consider a set of rigid pucks of varying mass (and possibly varying diameter) lying on a table. If it is an air-hockey table then the pucks are free to move in the plane according to Newton's law $\tau = m\ddot{x}$. If it is an un-lubricated workbench, assuming that the dynamics are dominated by friction, we might adopt a generalized damper model with dynamics $\tau = m\dot{x}$. We describe the construction of a force law for assemblies possessing either generalized damper or Newtonian dynamics which will (i) drive the pucks into a desired "assembly" while (ii) avoiding collisions between them. We will consider two extremes of actuator availability. First, suppose that each puck is equipped with an actuating device that allows us to instantly and independently command the force applied to it. The general results of the potential field approach apply directly to this case, and their application is treated in Sections 3.1 and 3.2. Applying these force-laws cause the pucks jostle their way past each other (without contact) into the "assembled" position much like

soldiers scrambling into parade formation, as is pictured in Figure 1. Of course, it is unreasonable to assume that every degree of freedom of the assembly (i.e. each puck) is independently actuated. In practice we wish to assemble, say, a pallet of stacked boxes (possessing perhaps hundreds of degrees of freedom) with a single robot manipulator (possessing just a few degrees of freedom). Section 3.1 also discusses, for the first order (generalized damper dynamics) case, a "sequentialized" controller which requires forces to be applied to just one disk at a time. In this "sequentialized" version, the individual force on the disks might be applied by a single robot arm, positioned above the workplace, which would achieve the desired assembly (without collision) by moving just one puck at a time. Preliminary simulations reported in Section 4 examine the performance of these algorithms.

Although there is unlikely to be any increase in computational efficiency arising from this approach, we do hope for at least two relative advantages in the context of assembly. First, within our framework, the problem specification is simply the natural geometric and dynamic properties of the parts. This data is incorporated symbolically into our controllers and there is growing evidence to support the hope that uncertainties (e.g. tolerances) or small changes requiring re-computation of more abstracted representations might be handled straightforwardly by recourse to symbolic "deformation" techniques [12]. Second, one would expect that our feedback-based strategies might offer the desirable stability and robustness properties characteristic of closed-loop controllers in contrast to their open-loop counterparts.

2 A Navigation Function for 2-D Sphere Assemblies

This section first describes a simple class of assembly tasks and then describes the construction of what we believe (but have not yet proven) to be a Navigation Function for this class of simple assembly tasks.

2.1 2-D Sphere Assemblies

Each of n balls, which are free to move, is uniquely specified by its position $b_i \in \mathbb{R}^2$, radius $\rho_i \in \mathbb{R}^+$, and the composite vector of n balls is $b \in \mathbb{R}^{2n}$. Label the *desired* ball positions $d_i \in \mathbb{R}^2$ and $d \in \mathbb{R}^{2n}$ respectively. Figure 1 shows both a typical 7 ball and 15 ball assembly sequence from (random) initial configuration to "assembled" final configuration.

2.2 What is a Navigation Function?

We desire a smooth function $\varphi : \mathbb{R}^{2n} \rightarrow [0, 1]$ from the configuration space of the assembly to the unit interval which (i) attains a uniform maximum value on the configuration space boundary (those configuration space points which correspond to workspace collisions) and (ii) has a *unique* global minimum at the desired point in configuration space, all other critical

*This work was supported in part by the National Science Foundation under a Presidential Young Investigator Award held by the second author.

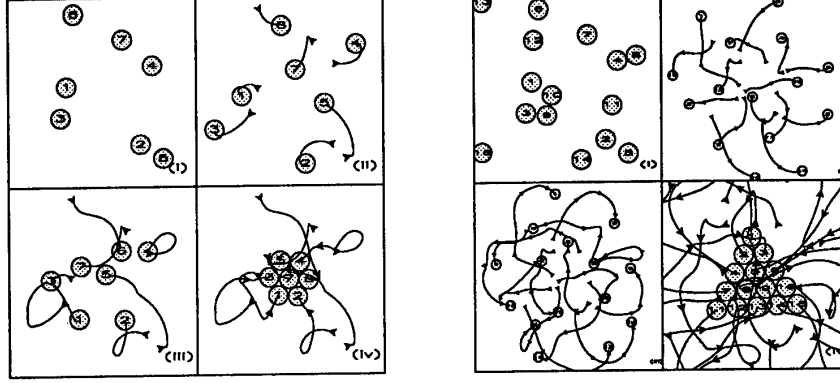


Figure 1: A 7 Ball Assembly Sequence (left) and A 15 Ball Assembly Sequence (right). Balls indicate current configuration b , Lines indicate the ball paths. Frame (i) shows random initial configuration, (ii) and (iii) are intermediate configurations, and (iv) shows final desired (assembled) configuration.

points being isolated saddles. With some additional technical conditions, this describes the class of Navigation Functions [6]. Given a function $\varphi(b)$ possessing these properties, the state of the gradient dynamical system

$$\dot{b} = -\nabla \varphi(b) \quad (1)$$

will asymptotically approach d , without collision, from all initial conditions *except* those within the basin of attraction of the (topologically inevitable) saddle points — a set of measure zero [6].

2.3 The Navigation Function

This section describes the construction of a family of functions clearly possessing the first property, but not necessarily possessing the second. Koditschek [5] has recently proven that property (ii) holds for a one-dimensional version of this problem. As the simulations in Section 4 will suggest, this construction *appears* to possess the property for the two-dimensional case. We surmise that the property holds in general.

First construct the cost function $\alpha(b) = \gamma(b)^k / \beta(b)$ where $\gamma(b)$ is a smooth function which is positive everywhere except at the point $b = d$ (where it has a value of zero), k is a design parameter, and $\beta(b)$ is a smooth function which is positive everywhere except at those configurations where a ball makes contact with another body (where it has value zero).

The function $\gamma(b)$ is given by

$$\gamma(b) = \sum_{i=0}^{i=n} \gamma_i(b); \quad \gamma_i(b) = \frac{1}{2} \|b_i - d_i\|^2 \quad (2)$$

and $\beta(b)$ is given by

$$\beta(b) = \prod_{\substack{i=n, j=n, i \neq j \\ i=1, j=1}} \beta_{b_i, b_j}(b) \quad (3)$$

where the $\frac{n}{2}(n-1)$ functions $\beta_{b_i, b_j}(b) = \frac{1}{2} (\|b_i - b_j\|^2 - (\rho_i + \rho_j)^2)$ are always positive when the balls are not touching, and

zero when ball i and j touch. We limit the magnitude of $\alpha(b)$ to the unit interval $[0, 1] \subset \mathbb{R}$ (and hence limit its gradient) by taking the composition with a “squashing” function

$$\varphi(b) = \sigma(b) \circ \alpha(b); \quad \sigma(q) = (q/(1+q))^{\frac{1}{k}}. \quad (4)$$

Taking the gradient of φ we have

$$\nabla_b \varphi = \left(\frac{1}{k} (\beta + \gamma^k)^{-\frac{1}{k} + 1} \right) [k\beta \nabla_b[\gamma] - \gamma \nabla_b[\beta]] \quad (5)$$

The vector-valued components of equation (5) is a sum of two components each deserving careful consideration. The first, $k\beta \nabla_b[\gamma]$, is “attractive” — pointing in the direction of the desired destination in configuration space, and is scaled both by the design parameter k and the function β . The second component, $\gamma \nabla_b[\beta]$ is “repulsive” — pointing “away” from collisions, and is scaled by the function γ . For a formal treatment of these issues the reader is referred to [6].

3 Feedback Control of 2-D Sphere Assemblies.

In this section we demonstrate how the navigation functions defined in Section 2 may be employed to construct feedback laws for the control of 2-D sphere assemblies. For proofs of the asymptotic stability and safety of these systems the reader is referred to [4, 14]. In the sequel we will, without loss of generality, employ $M = I$ for simplicity.

3.1 Feedback Control of Assemblies with First Order Dynamics

In this section we will adopt the simple first order generalized damper model $\tau = \dot{x}$ for the motion of 2-D sphere assemblies, and will propose three feedback controllers (force laws) which will drive the balls into a desired “assembled” configuration while avoiding collisions.

3.1.1 The Gradient Controller

We conjecture that $\varphi(b)$ (4) is a Navigation Function. If so, then according to our previous work [6], the closed loop

system resulting from the controller

$$\tau = -\nabla\varphi(b) \quad (6)$$

results in a gradient dynamical system identical to (1) whose trajectories define collision free paths to the desired “assembled” configuration-space coordinate d .

3.1.2 The Normalized Gradient Controller

In practice the magnitude of $\nabla\varphi$ is often small — values smaller than 10^{-20} are commonly observed even in simple cases. These small gradient norms present two difficulties. First, since the “speed” of the gradient system (1) is precisely $\|\nabla\varphi\|$, the system may take an extraordinary amount of time to converge to the desired point d . Second, numerical integration of (1) (which requires the addition of quantities which scale with $\|\nabla\varphi\|$ to quantities of order $\|b\|$) is limited by machine numerical precision¹ — and can fail.

To remedy this problems, we can normalize the controller (6) by taking its product with the scalar valued function

$$\lambda = (\epsilon + \|\nabla\varphi\|)^{-1}; \quad \epsilon > 0. \quad (7)$$

where ϵ is empirically selected. This results a controller of the form

$$\tau = -\lambda\nabla\varphi(b) \quad (8)$$

and gives rise to the closed loop dynamical system

$$\dot{b} = g(b) = -\lambda \cdot \nabla\varphi(b) = -\nabla\varphi(b)/(1/(\epsilon + \|\nabla\varphi\|)) \quad (9)$$

which is free from the performance problems of (1) described above yet can be shown to inherit its correctness properties.

3.1.3 The Sequential Normalized Controller

In practice, it is not possible for a single robot arm to *simultaneously* and *independently* manipulate, say, a dozen workpieces. A robot manipulator is typically capable of grasping and manipulating one object at a time. We now suggest how the continuous feedback laws developed above may be adapted to such a situation.

Sequential Algorithm – Low Level Part: Define n low level controllers to be simply the projection of the normalized gradient controller (8) onto the i^{th} subspace \mathbb{R}^2 associated with the i^{th} ball

$$\tau = f_i(b) = S_i g(b); \quad i \in \{1, \dots, n\} \quad (10)$$

where $S_i = [\text{Diag}(0, 0, \dots, 1, \dots, 0) \otimes I_{2 \times 2}]$. For the first order plant this results in the closed loop dynamical system $\dot{b} = -S_i \nabla\varphi(b)/(\epsilon + \|\nabla\varphi\|)$.

Sequential Algorithm – High Level Part: It is the job of the *high level* algorithm to sequentially choose from among the i low level differential equations, evaluate the solution until it “blocks”, and then select a different one. With some abuse of notation², we can write the high level controller as the discrete dynamical system

$$y_{n+1} = T(y_n) \quad (11)$$

¹For example, in the standard double length IEEE floating point representation, for x smaller than about 10^{-16} , $1.0 + x = 1.0$

²Here we use $y \in \mathbb{R}^{2n}$ to represent the configuration space state of the high level controller at the i^{th} sequential step to distinguish it from $b_i(t)$, the x - y position of the i^{th} ball at time t .

where the $T : \mathbb{R}^{2n} \mapsto \mathbb{R}^{2n}$ is the transition map from one “blocked” ball configuration to the next. T is given by

$$\begin{aligned} T(y_n) &= \Omega_{i_n}(y_n); \quad i_n \equiv C(y_n) \\ \Omega_i(y) &= \lim_{t \rightarrow \infty} \{\theta(t) : \dot{\theta} = f_i(\theta); \theta_0 = y\} \end{aligned} \quad (12)$$

where $\Omega_i(y)$ is function returning the limit point of the i^{th} first order system with initial condition y . Finally, the selection function $C(y)$ returns the index of that $f_i(y)$ (10) with the greatest magnitude [14].

3.2 Feedback Control of Assemblies with Second Order Dynamics

In this section we will adopt the simple second order dynamical model $\tau = \ddot{x}$ for the motion of the 2-D sphere assemblies where τ is the vector of forces applied to the 2-D disks, and \ddot{x} the vector of resulting ball accelerations. Rewriting the second order plant (13) as a system of first order equations we have

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \tau \end{aligned} \quad (13)$$

We propose two feedback controllers (force laws) which will drive the balls into a desired “assembled” configuration $x_d = [x_1^T, x_2^T]^T = [d^T 0^T]^T$ while avoiding collisions.

3.2.1 Constructing a Feedback Controller for a Second Order System from the Gradient System

Let us again in this section consider φ rewritten in the error coordinate system $[e_1^T, e_2^T]^T = [(x_1 - x_1^*)^T, (x_2 - x_2^*)^T]^T$ which has x_d at its origin.

We first propose the following simple feedback controller

$$\tau = -[\nabla\varphi(e_1) + K_2 e_2]. \quad (14)$$

The term $\nabla\varphi(e_1)$ provides the feedback on position error. K_2 is a positive definite symmetric matrix, thus $K_2 x_2$ is a simple damping term providing feedback of velocity error. This results in the closed loop system

$$\dot{e} = L(e); \quad L(e) \triangleq \begin{bmatrix} -(\nabla\varphi(e_1) + K_2 e_2) \end{bmatrix}, \quad (15)$$

an example of a dissipative Lagrangian system whose properties we have discussed at length in more tutorial and more formal presentations [4]. We have shown that this “generalized spring-mass-damper” system is correct with respect to the navigation task. The closed loop system converges from almost every initial condition, $(x_1(0), x_2(0))$, with bounded total energy, $x_2(0)^T x_2(0) + \varphi(x_1(0)) \leq 1$, to the desired destination, x_d . Moreover, no initial condition with bounded total energy as indicated is associated with any collisions between the bodies along the way to the final destination. Thus, the lifted controller inherits the safety (all initial conditions satisfying a “speed limit” that depends upon the maximal amount of input torque or force available have trajectories that avoid obstacles) and convergence (almost every safe initial condition has a trajectory that arrives at the desired destination) properties of the first order gradient scheme (6).

3.2.2 Constructing a Feedback Controller for a Second Order System from the Normalized Gradient System

The controller of Section 3.2.1 for the second order plant (13) suffers from the same defect observed for the first order

controller of Section 3.1.1 with its first order plant. Here again, the un-normalized algorithm is “correct”, yet yields unacceptably slow convergence. In this section we address this defect by “lifting” the first order controller of Section 3.1.2 to the second order system.

In this section we employ the error coordinates

$$\begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} x_1 - x_1^* \\ x_2 - g(x_1) \end{bmatrix} \quad (16)$$

where g is defined in (9). Notice that the desired point x_d again corresponds to the origin of the new coordinate system. Now consider the control law,

$$\tau = -\nabla\varphi(e_1) - K_2 e_2 + \dot{g}; \quad \dot{g} \triangleq Dg(e_1) e_2, \quad (17)$$

resulting in a closed loop system expressed in the present coordinates as

$$\begin{aligned} \dot{e}_1 &= e_2 + g(e_1) \\ \dot{e}_2 &= -(\nabla\varphi(e_1) + K_2 e_2). \end{aligned} \quad (18)$$

For a proof of the asymptotic stability and safety of this system the reader is referred to [4, 14].

4 Simulations

In this section we present the results of computer simulation of the closed loop behavior of the first and second order systems. For this initial investigation we have employed perhaps the simplest and most easily measured performance metric — assembly path-length. Assembly path-length is the distance (in \mathbb{R}^{2n}) traveled by the spheres from an initial configuration to a desired final “assembled” configuration. Since path-length necessarily varies with initial condition, we wish to normalize results with respect to the minimum attainable path-length. However, since minimum attainable path-length is difficult to compute in practice, we have instead normalized the results with respect to the straight-line euclidean distance from the initial to final point, and shall term this measure the “path-length ratio” in the figures³.

4.1 Performance of the First Order Assembly System

In this section we first examine two performance aspects of the normalized gradient assembly system (1).

The Effect of the Parameter K on Pathlength: Earlier work on a point moving amongst sphere obstacles concludes that φ , (4), becomes a navigation function when k is sufficiently large. We have observed in simulation (and are in the process of proving rigorously) that values of k above a context-defined threshold ensure solutions to (1) converge to the desired point d . Conversely, we have also observed solutions to (1) do not converge to d for values of k below this threshold. Figure 2 shows the path-length ratio at five different values of k for the seven ball assembly of Figure 1⁴.

³Note that the path-length ratio for the unswitched algorithm is with respect to the configuration space euclidean norm from initial to final configuration, whereas the the switched version is computed with respect to the sum of the individual ball distances.

⁴Each data point of every graph in this paper represents the Mean and Standard Deviation of 25 runs with random initial configuration.

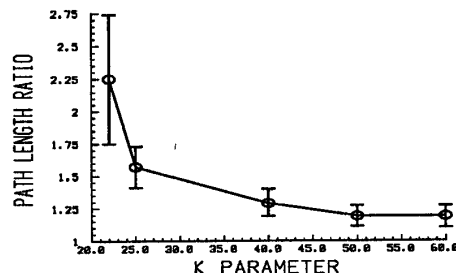


Figure 2: First Order System: Path Length Ratio⁴ vs k for the 7 ball assembly of Figure 1.

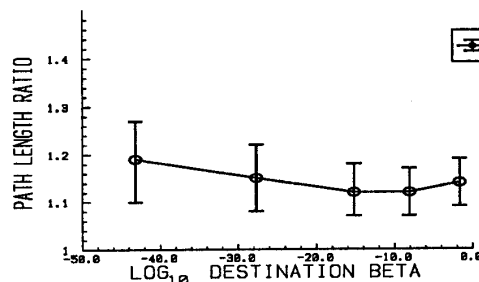


Figure 4: First Order System: Path Length Ratio⁴ vs $\beta(d)$ for the assemblies of Figure 3.

Higher values of k clearly yield smaller mean path-length ratios and, from the proportionally smaller standard deviations we may conclude this relationship is “typical”. In this example we see mean path-length ratios as low as the 1.25 (for the highest k value), indicating that solutions to (1) have average length only 25% greater than the optimum path. Indeed, we have observed similar performance figures for a great variety of assemblies.

The Effect of Assembly “Difficulty” on Path-length: We naturally expected path-length performance to vary with the “difficulty” of the assembly task — that the closely packed desired assembly of Figure 3(d) would be more difficult to attain than a loosely packed assembly of figure 3(a). Figure 3 shows that $\beta(d)$ varies in a manner that corresponds to our intuitive notion of assembly difficulty —

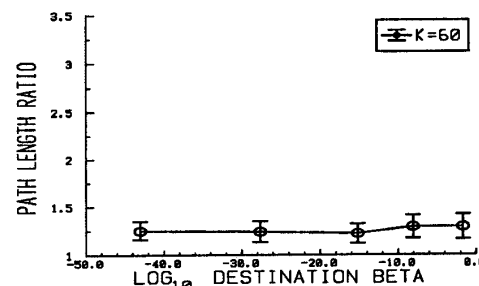


Figure 5: First Order Switched Algorithm System: Path Length Ratio⁴ vs $\beta(d)$ for the assemblies in Figure 3.

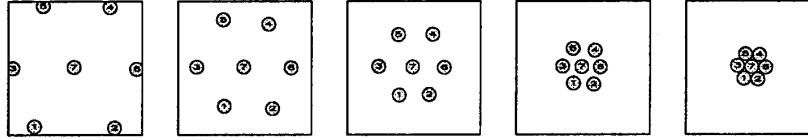


Figure 3: Five 7-Ball Assemblies of Varying Difficulty: (l to r) (a) $\beta(d) = 10^{-2}$, (b) $\beta(d) = 10^{-9}$, (c) $\beta(d) = 10^{-16}$, (d) $\beta(d) = 10^{-28}$, (e) $\beta(d) = 10^{-44}$.

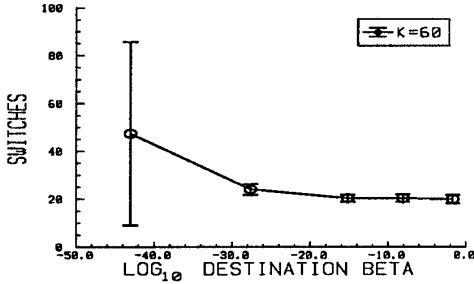


Figure 6: First Order Switched Algorithm System: Switch Count⁴ vs $\beta(d)$ for the assemblies in Figure 3.

The tightly packed assembly of Figure 3(d) has $\beta(d)$ value of 10^{-44} while an “easier” loosely packed assembly of 3(a) has $\beta(d)$ value 10^{-2} . Figure 4 shows the resulting plot of path-length ratio for five different values of destination beta. Surprisingly, the path-length ratio appears to be nearly independent of the assembly $\beta(d)$ value. “Harder” assemblies result in path-length ratios only marginally greater than for “easy” assemblies.

Performance of the Switched Gradient Assembly System: Figure 5 shows the means and standard deviations of the switched algorithm’s path length ratios at five different $\beta(d)$ values. This figure represents the switched version of the data portrayed in Figure 4. A comparison of Figure 4 (unswitched) and Figure 5 (switched) reveals essentially similar performance, with the switched algorithm path-length ratio only 10% greater than that of the unswitched.

Figure 6 shows the corresponding mean and standard deviations for the *number of switches* for the runs of Figure 5. Here we observe that the number of switches required to complete an assembly rises perceptibly as a function of the assembly difficulty. The “easy” assemblies here require each ball to be switched (grasped, moved, and released) only about three times, while the “difficult” assemblies have both a higher mean number of switches as well as substantially higher variance.

4.2 Performance of the Second Order Assembly System

In this section we examine the performance aspects of the normalized second order assembly system (18).

The Effect of the Parameter K on Pathlength: Figure 7 shows the path-length ratio as a function of k for the 7 ball second order assembly corresponding to Figure 1. The data for this figure was generated with $K_d = 0.01$. Here

again, low values of k resulted in relatively poor performance, and high values of k resulted in excellent performance. For higher values of k (for which the first order system was typically within 25% of optimum) the performance of the second order system to be typically within 50% of optimum.

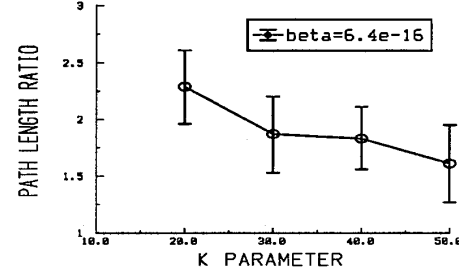


Figure 7: Second Order System: Pathlength Ratio⁴ vs K Parameter for assembly of Figure (3c).

The Effect of Assembly “Difficulty” on Path-length: Figure 8 shows a plot of path-length ratio for five different values of destination beta. Figure 8 echoes that of the first order system, Figure 4, again demonstrating that the remarkably good path-length performance reported in the previous section is observed to hold for both easy (large $\beta(d)$) and hard (small $\beta(d)$) assemblies.

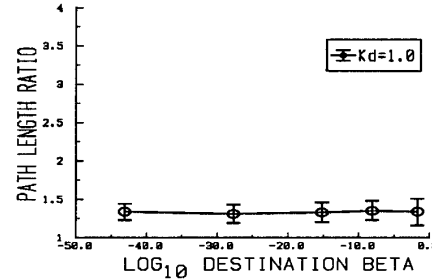


Figure 8: Second Order System: Path Length Ratio⁴ vs $\beta(d)$ for the assemblies in Figure 3.

The Effect of K_d Velocity Damping on Path-length: The second-order system has a free “damping” parameter, K_d , which has no counterpart in the first order assembly controller. In the experiments of this section we have used K_d values which are scalar multiples of the identity matrix. Figure 9 shows the second order system path-length

ratio as a function of K_d . A k parameter value of 30 was used. Path-length performance is seen to be strongly dependent on K_d . Low values of K_d produce long path-lengths, and high values of K_d produce short path-lengths. The highest values of K_d are seen to typically result in path-length performance within 50% of optimum. The effect of increasing the K_d parameter value is, roughly speaking, to "force" the second order system onto the stable first order manifold $\dot{x}_1 = g(x_1)$. Qualitatively, high K_d values produce "overdamped" trajectories and low K_d values produce "underdamped" oscillatory trajectories.

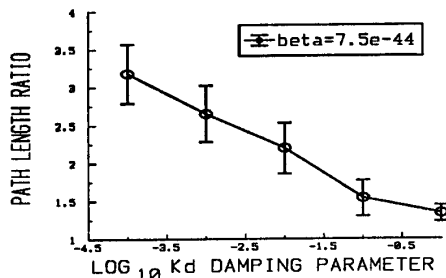


Figure 9: Second Order System: Pathlength Ratio⁴ vs K_d Damping Parameter for the assembly depicted in Figure 1.

The Effect of ϵ Normalization Parameter on Pathlength: The scaling parameter ϵ is employed in the normalized version of the planning gradient (8). In practice we found it often impossible to find a single value of ϵ both (i) small enough to provided satisfactory fast convergence times and (ii) large enough to the avoid numerical problems. The second order simulations, then, were run with a simple ad-hoc switched ϵ . Beyond an arbitrarily defined distance from the desired assembly x_d , the simulations employed $\epsilon = 0$ to ensure good convergence times. Within the distance threshold, the simulation switched to use $\epsilon = 10^{-6}$ to provide a smooth vector field near the destination x_d . While the introduction of a nonzero ϵ is of significant theoretical importance (the proofs hold only for $\epsilon > 0$) we surmise that the use of $\epsilon = 0$ will provide satisfactory performance in actual practice. The matter deserves careful attention.

5 Conclusion

We have presented an approach to automatic assembly that holds the promise of providing a control scheme capable of effecting the physical motions needed to perform the assembly. We have explored the performance of a concrete instance of this approach in a very simple problem setting — re-arranging disks on a plane into specified finished patterns. We have presented force laws for case that the disks possess simple first or second order dynamics. We have extended the control theory surrounding the gradient based versions of this approach to the case of non-gradient based plans.

Simulations suggest that the obtained configuration space paths that are *typically* (in the statistical sense of their path length ratio mean and standard deviation for repeated trials with random initial configurations) better than within a factor of two of the optimal length. Performance varies with a design parameter (k) whose proper adjustment can apparently deliver configuration space paths that are typically within 25% of the optimal length for first-order plants, and

within 50% for second-order plants. Somewhat surprisingly, performance appears to vary little with assembly "difficulty".

A "sequential" assembly controller which requires only a single body to be manipulated at a time was presented for the case of disks possessing first order dynamics. Simulations of this "sequential" system demonstrated path length ratios roughly 10% higher than those corresponding to the previous systems requiring simultaneous motion of all n bodies. The number of "switches" in these cases appears to increase with assembly "difficulty" and to decrease with increasing k . In an n body assembly plan sequence for a single robot, the number of switches in the cases of even greatest "difficulty" is typically less than n per body.

Acknowledgments

We thank Dr. Elon Rimon for a number of valuable discussions concerning this problem.

References

- [1] B. R. Donald and P. G. Xavier. Time-safety trade-offs and a bang-bang algorithm for kinodynamic planning. In *IEEE International Conference on Robotics and Automation*, pages 552–557, Sacramento, CA, USA, 1991.
- [2] L. S. Homem de Mello and A. C. Sanderson. A correct and complete algorithm for the generation of mechanical assembly sequences. *IEEE Transactions on Robotics and Automation*, 7(2):228–240, April 1991.
- [3] O. Khatib. Real time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90–99, Spring 1986.
- [4] D. E. Koditschek. The control of natural motion in mechanical systems. *ASME Journal of Dynamic Systems, Measurement, and Control*, 113(4):547–551, Dec 1991.
- [5] D. E. Koditschek. An approach to autonomous robot assembly. *Robotica*, (to appear), 1992.
- [6] D. E. Koditschek and E. Rimon. Robot navigation functions on manifolds with boundary. *Advances in Applied Mathematics*, 11:412–442, 1990.
- [7] J.-C. Latombe. *Robot Motion Planning*. Kluwer, Boston, MA, 1991.
- [8] F. Miyazaki and S. Arimoto. Sensory feedback based on the artificial potential for robots. In *Proceedings 9th IFAC*, Budapest, Hungary, 1984.
- [9] W. S. Newman and N. Hogan. High speed robot control and obstacle avoidance using dynamic potential functions. In *Proc. IEEE International Conference on Robotics and Automation*, pages 14–24. IEEE, Raleigh, NC, 1987.
- [10] V. V. Pavlov and A. N. Voronin. The method of potential functions for coding constraints of the external space in an intelligent mobile robot. *Soviet Automatic Control*, (6), 1984.
- [11] E. Rimon. A navigation function for a simple rigid body. In *IEEE International Conference on Robotics and Automation*, pages 546–551, Sacramento, CA, USA, 1991.
- [12] E. Rimon and D. E. Koditschek. The construction of analytic diffeomorphisms for exact robot navigation on star worlds. *Transactions of the American Mathematical Society*, 327(1):71–115, Sep 1991.
- [13] E. Rimon and D. E. Koditschek. Exact robot navigation using artificial potential fields. *IEEE Transactions on Robotics and Automation*, (to appear), 1992.
- [14] L. L. Whitcomb, D. E. Koditschek, and J. B. D. Cabrera. Toward the automatic control of robot assembly tasks via potential functions: The case of 2-d sphere assemblies. Technical report, Department of Electrical Engineering, Yale University, 1992.