This document contains the draft version of the following paper:

A.G. Banerjee, A. Pomerance, W. Losert, and S.K. Gupta. Developing a stochastic dynamic programming framework for optical tweezers based automated particle transport operations. *IEEE Transactions on Automation Science and Engineering*, 7(2), 218 – 227, 2010.

Readers are encouraged to get the official version from the journal's web site or by contacting Dr. S.K. Gupta (skgupta@umd.edu).

Developing a Stochastic Dynamic Programming Framework for Optical Tweezer based Automated Particle Transport Operations

Ashis Gopal Banerjee, *Student Member*, *IEEE*, Andrew Pomerance, Wolfgang Losert, and Satyandra K. Gupta

***Abstract*— Automated particle transport using optical tweezers requires the use of motion planning to move the particle while avoiding collisions with randomly moving obstacles. This paper describes a stochastic dynamic programming based motion planning framework developed by modifying the discrete version of an infinite-horizon partially observable Markov decision process algorithm. Sample trajectories generated by this algorithm are presented to highlight effectiveness in crowded scenes and flexibility. The algorithm is tested using silica beads in a holographic tweezer set-up and data obtained from the physical experiments are reported to validate various aspects of the planning simulation framework. This framework is then used to evaluate the performance of the algorithm under a variety of operating conditions.**

Note to Practitioners— **Micro and nano scale component-based devices are revolutionizing health care, energy, communication, and computing industry. Components need to be assembled together to create useful devices. Such assembly operations remain challenging in spite of the advancements in imaging, measurement, and fabrication at the small scales. This paper deals with directed assembly using optical fields that is useful for prototyping new design concepts, repairing devices, and creating templates for self-assembly.**

***Index Terms*— Stochastic dynamic programming, partially observable Markov decision process, optical tweezer, microsphere, automated planning, simulation**

I. Introduction

Strongly focused light beams exert optical gradient and scattering forces on objects; using this property of light beams, optical tweezers (OT) have been developed to successfully trap, orient and move nanoscale and microscale components of many different sizes and shapes [1]-[3]. Components can be simply released from optical traps by switching off the lasers. Thus, OT overcome some of the challenges associated with contact based grippers at the microscale. Two types of multipoint tweezer set-ups are commonly used: scanning mirror based [4] and holographic [5], each with its own advantages and limitations. While the former type can control the traps at a faster speed, the latter is able to apply forces continuously and generate a larger number of traps quite easily.

Our aim is real-time motion planning for automated assembly using OT. This involves trapping the desired objects (*target objects*), transporting them to user-specified goal locations and avoiding collisions with other objects (*obstacles*) present in the workspace if possible. This is similar to robot motion planning problems that have received a lot of attention over the past few decades. Broadly, the methods can be classified into two types: planning with complete information or with incomplete information. Inspired by the Piano Mover's problem, a large number of problems have been studied in the former domain [6]. Many of the approaches utilize the notion of *configuration space* (*C-space*) [7]. A detailed description of the popular methods in this category, including visibility graph, road map, potential field etc. can be found in [8]. On the other hand, the latter class of problems has been studied only in the recent past. A detailed survey of approaches to solve these problems can be found in [9]. A chronological review of the existing classical and heuristic approaches can be found in [10].

Our problem has some unique characteristics due to the small size scales involved. The environment is constantly changing due to the random, Brownian motion of the objects. In addition, OT attracts objects into the laser trap centers even as far away as few microns; so care must be taken to maintain sufficient separation from all the obstacles. By utilizing synchronized time sharing or dynamically created holograms, multiple optical traps can manipulate several objects in parallel. Hence, additional traps can be used to keep nearby obstacles stationary by trapping them. Due to the uncertainty in identifying object positions using an imaging-based sensor, and the time delay between identification and trap placement,

Manuscript received November 19, 2008. This research has been supported by NSF grants CMMI-0835572 and OCI-0636164. Opinions expressed in this paper are those of the authors and do not necessarily reflect those of the sponsors.

Ashis Gopal Banerjee is with the Department of Mechanical Engineering and the Institute for Systems Research, University of Maryland, College Park, MD 20742 USA (e-mail: ashis@umd.edu).

Andrew Pomerance is with the Department of Physics, University of Maryland, College Park, MD 20742 USA (e-mail: pomeranc@umd.edu)

Wolfgang Losert is with the Department of Physics, Institute for Physical Science and Technology, and Institute for Research in Electronics & Applied Physics, University of Maryland, College Park MD 20742 USA (e-mail: wlosert@umd.edu).

Satyandra K. Gupta is with the Department of Mechanical Engineering and the Institute for Systems Research, University of Maryland, College Park, MD 20742 USA (phone: 301-405-5306, fax: 301-314-9477, e-mail: skgupta@umd.edu).

the event of trapping an object has a non-zero probability of failure associated with it. This suggests that we need to utilize the approaches used in the domain of probabilistic robotics [11], [12].

Primarily, two types of approaches are quite popular – dynamic programming and probabilistic roadmaps. The former approach has been combined with the notion of classical preimages [8] to form an objective framework for planning under both control and sensing uncertainties [13]. It has also been incorporated within a unified, game-theoretic framework for design and analysis of algorithms to solve a wide variety of problems in [14]. A discrete version of the infinite-horizon dynamic programming has been applied to steer flexible bevel-tip needles inside soft tissues in [15]. A two-step (consisting of learning and query phases) probabilistic roadmap approach was first used for planning in higher-dimensional *C*-spaces in [16]. More recently, it has been used for real-time path planning of multiple virtual agents in dynamic environments in [17]. A belief-space variant of the probabilistic roadmap algorithm, called the belief roadmap, has been developed in [18] by using a factored form of the covariance matrix associated with linear Gaussian systems.

An up-to-date review of both geometric and probabilistic motion planning approaches, along with techniques for position estimation and mechanical control systems can be found in [19]. Lot of work has also been carried out on replanning in unknown and dynamic environments. Various types of rapidly-exploring random trees (a variant of probabilistic roadmap) have been used in [20], [21]. A greedy but probabilistically complete state space exploration strategy for replanning under kinodynamic constraints in non-holonomic systems using incrementally updated tree data structure has been presented in [22].

It may be noted here that, unlike many other problems, the time available for planning in our case is limited by the imaging interval (Δt_i), image processing time (t_d), controller update interval (Δt_c), and the continuous random motion of all the non-trapped objects. Ideally, the planning time should be very short as compared to both Δt_i and Δt_c such that the random motion does not change the obstacle positions so much that the plan is rendered ineffective. This paper develops a stochastic dynamic programming framework to perform the above-mentioned planning operation.

II. Problem Formulation

A 3D rectangular parallelepiped shaped assembly cell set-up is considered as our *workspace W*. It is assumed that the laser beam propagates upward along the +*z*-axis. Currently, OT systems are primarily used to manipulate spherical objects (glass, amorphous silica, polystyrene, biological cells etc.). Hence, only spherical objects of identical size are handled in this paper. We will only consider transporting objects in the same horizontal plane. The physics of the problem is modeled as follows:

- Any object in the vicinity of a laser beam in the workspace experiences five forces, namely, optical trapping, Langevin/thermal (causing Brownian motion), viscous drag, gravity, and buoyancy.
- A normal distribution based position sampler is used to generate random, Brownian positions of any object at discrete time intervals. Mean of the distribution is time-invariant for every individual object, whereas the standard deviation is equal to the diffusion length.
- Random position sampling is combined with the ray optics model for trapping forces (given in [1]), drag term given by Stokes' law, and well-known constant expressions for gravity and buoyancy to generate a list of positions, and velocities of the objects. It should be pointed out here that the trapping force values obtained from the ray optics model are suitably reduced based on the experimental data available for smaller-sized beads that are close to 1 μm in diameter ([23], [24]).
- Diffusion length of an object in 3D as a function of time *t* is expressed as $r = \sqrt{6Dt}$, where the diffusion coefficient (*D*) is given by Einstein-Smoluchowski relation to be $D = \frac{k_B T}{\gamma}$. Here k_B is the Boltzmann constant and γ is the viscous drag coefficient. From Stokes' law, $\gamma = 6\pi\eta R_a$, where R_a is the radius of the object and η is the viscosity of the fluid medium. It may be noted here that the diffusion length along each dimension is independent of one another and is equal to $\sqrt{2Dt}$.
- Sensor uncertainty or measurement probability is modeled using independent normal distributions (having identical standard deviations σ) along the coordinate axes.
- Collisions are modeled as being hard and elastic. Only pair-wise collisions between objects are considered and chain collisions are ignored. This works well due to the small time interval at which object positions are calculated.
- If multiple objects are trapped by a single laser beam, then they collide with each other and consequently either both of them may get trapped (smaller-sized objects) or none of them may remain trapped (larger objects).
- For relatively large objects (diameter more than 10 μm), the maximum permissible trap speed ($v_{\max}$) is obtained by equating the viscous drag force with the maximum optical trapping force. However, the Brownian motion term dominates for smaller-sized objects. In that case, $v_{\max}$ is obtained by performing simulation or physical experiments.
- If multiple traps are switched on, then $v_{\max}$ is reduced proportionately.

For planning purposes, we have defined t*rapping probability* that refers to the probability of trapping an object that lies in a spatial region close to the optical trap center. Based on the dynamics simulation reported in [25], it has a value of 1 inside a horizontal circle centered at the laser beam focus with radius equal to trap radius (l_{tr}), is equal to 0

outside a concentric circle with radius equal to the sum of trap radius and trap width ($l_{tr}+l_{tw}$), and varies linearly as a function of distance within the annular region, determined by l_{tr} and ($l_{tr}+l_{tw}$). The radius of effective obstacle region (centered at the current, estimated object position) arising due to presence of optical trap, finite size, and imaging uncertainty is then given by: $r_{Sys}=\max\left\{\left(\left(l_{tr}+l_{tw}\right)+3\sigma\right),\left(2R_a+3\sigma\right)\right\}$.

Actual locations of all the objects in the workspace form the true workspace set, whereas the estimated locations form the estimated workspace set. The primary trap is solely used to transport the target object, whereas the secondary traps are utilized to keep the obstacles stationary. Mathematical definitions of these concepts are given below:

Definitions

True workspace state is the set $X_t=\left\{X_1^t,\ldots,X_n^t\right\}$ where X_i^t is the actual coordinate $\left(x_i^t,y_i^t,z_i^t\right)$ of the center of the i^{th} object in *W*. Without any loss of generality, the first object can always be considered as the *target object* and the remaining (*n*-1) can be treated as *obstacles*.

Estimated workspace state is the set $X_e=\left\{X_1^e,\ldots,X_n^e\right\}$ where X_i^e is the estimated coordinate $\left(x_i^e,y_i^e\right)$ of the center of the i^{th} object in *W*. The estimated z-locations of the object centers are only used in Section V. The planner makes all decisions based on X_e and not X_t. It is useful to point out here that all the objects may not be detected at all the times due to limitations in the imaging hardware and feature recognition modules.

Control action set

- **Primary** – A 3-tuple $\left\{s,\left(f_x,f_y\right),\left(v_x,v_y\right)\right\}$, where *s* is a binary integer corresponding to the state of the *primary trap* (1 if on and 0 if off), $\left(f_x,f_y\right)$ represents the coordinate of the trap center (beam focus) and $\left(v_x,v_y\right)$ is the uniform velocity vector assigned to the trap.
- **Secondary** - A set of 9 doubles $\left\{\left\{s_1,\left(f_x^1,f_y^1\right)\right\},\ldots,\left\{s_9,\left(f_x^9,f_y^9\right)\right\}\right\}$ such that the 1st element of every double is a binary integer that denotes whether the corresponding *secondary trap* is switched on and $\left(f_x^i,f_y^i\right)$ represents the coordinate of the i^{th} trap center.

Objective function

A motion planning approach can handle collisions by ignoring them and re-trapping the target object, or by avoiding them either by circumventing obstacles, or by keeping the obstacles stationary using secondary traps. This is illustrated in Fig. 1 and forms the basis of our objective function.

Estimated, expected total transport time T_e for an estimated workspace state X_e, a control action *A,* and a circular circumvention strategy is defined as follows:

$$T_e\left(X_e,A\right)=\min\left\{\begin{array}{l}\left(\Delta t_c+\dfrac{\overline{F'G}}{v_{\max}}+\sum_{i=1}^{n_c}p_i t_{avg}\right),\left(\Delta t_c+\dfrac{\overline{F'G}}{v_{\max}}+\dfrac{\pi n_c r_{sys}}{v_{\max}}\right),\\ \left(\Delta t_c+\left(\dfrac{\overline{F'G}}{v_{\max}}\right)\left(1+\dfrac{wn_c\Delta t_c}{t_e}\right)+\left(1-w\right)\sum_{i=1}^{n_c}p_i t_{avg}\right)\end{array}\right\}$$

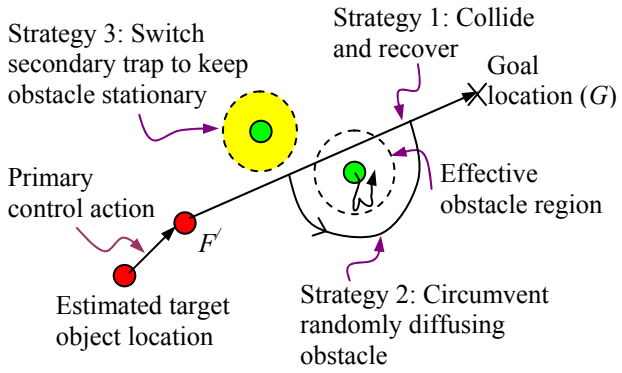


Fig. 1. Different collision avoidance strategies

The three terms in the above expression represent the estimated transport time using the three different collision handling strategies shown in Fig. 1. The first term allows collisions to occur and then imposes a penalty by introducing the additional time taken to recover the target object. The second term explores the option of circumventing all the freely diffusing obstacles and the third term considers the possibility of avoiding collisions by trapping obstacles to prevent them from moving randomly, which decreases the maximum transport speed.

$F'=\left(f_x\hat{i}+f_y\hat{j}\right)+\left(v_x\hat{i}+v_y\hat{j}\right)\Delta t_c$ is the new location of the primary trap center after Δt_c. n_c is the total number of expected collisions with obstacles based on the current X_e as the primary trap first moves with velocity $\vec{v}=v_x\hat{i}+v_y\hat{j}$ and then along $F'G$ (see Fig. 1). Such objects are termed as *potentially colliding obstacles*. p_i is the trapping probability of the i^{th} potentially colliding obstacle. t_{avg} is the average, expected time to re-trap an object by positioning it at the displaced location or switching the laser off, allowing the objects to drift and then turning it back after collision. *w* is the *expected* probability of avoiding collision by keeping any obstacle stationary. This is not equal to 1 as sometimes collision is inevitable because the trap may pass through the effective obstacle region and keeping it fixed does not help. t_e is the estimated time to reach *G* from the current location if the primary trap moves at the maximum speed. The above expression is easy to compute and is valid for circular circumvention. However, the actual circumvention path may vary from one approach to the other. A sequence of control actions is considered *optimal* if the estimated, expected total transport time is minimized.

III. MOTION PLANNING APPROACH

A. Algorithm Description

The motion planning problem under uncertainty is modeled as a Markov Decision Process (MDP). Since two types of uncertainties are present here, namely control action and measurement, it belongs to the class of partially observable Markov decision process (POMDP). This is also known as hidden Markov model or dynamic Bayes network. It utilizes the well-known Markov assumption that the past and future states are independent, provided the current state is known or estimated. Although inaccuracies in modeling the state transition and measurement probability may violate this assumption, Bayes filters used to derive the *belief state* is quite robust to such violations [12]. By belief state, we mean the posterior probability distribution of the current true state based on all the past estimated states and control actions.

POMDP algorithms require a *planning horizon* and an *expected cumulative payoff function* or *value function*. This function sums values over a number of time steps (T). If T is infinite, we obtain the *infinite-horizon approach*. It uses the *value iteration* formulation of the *Bellman equation* (Step 3.a.i.3 in our algorithm pseudo-code) to ensure optimality of the selected control action. It initializes the value function and then recursively propagates information in reverse temporal order until the Bellman equation converges. However, the summation inside the value function is performed over the belief state instead of the estimated state. We have used the discrete version of an approximate POMDP technique, known as QMDP, due to its ease of implementation and speed-up obtained over a full POMDP solution. In fact, QMDP is a hybrid between MDP and POMDP and is of the same complexity as MDP [12] based on the assumption that the state is fully observable after one step of control. It generalizes the MDP-optimal value function defined over the estimated state into a POMDP-type value function defined in the belief state by computing the so-called Q function (Step 4.a in the pseudo-code) using the *converged* value function. This Q function is then utilized to select the control action that yields the minimum expected value (Step 6).

The state transition matrix is computed once for all possible locations of the target object center and control actions, and stored in the local memory so that it can be used every time the planner is invoked. Since we only have a single objective function in this case, the problem remains tractable. A rectangular grid with a uniform grid spacing along both the orthogonal directions is overlaid on W. The estimated coordinates of the target object and the obstacle centers are approximated by the nearest grid point. Based upon pilot trial runs, a constant probability value of 0.9 is assigned to the expected grid location of the target object corresponding to the given control action. Values of 0.025 are assigned to each of the four other neighboring grid points. Belief function values are randomly sampled from the normal distribution representing the measurement uncertainty, multiplied with the predicted values (Step 4.b), and are normalized so that they all add up to 1 (Step 5).

Certain important modifications have been made to the original QMDP algorithm to adapt it for the current application. Firstly, the payoff function does not merely represent the one-step gain associated with that particular control action. Instead, it takes into account the overall gain corresponding to the given control action. Thus, it is similar in nature to the objective function and, consequently to the value function. This makes the iteration equation non-linear with respect to the distance (estimated, expected transport time in this case) of the goal from the current state and enhances convergence rates.

Secondly, time is introduced as an additional parameter within the convergence loop of the algorithm. This is necessitated by the fact that certain planning options that were optimal at the beginning of the planning time, may lead to inferior solutions with the passage of time and vice versa. Since all the non-stationary obstacles are exhibiting Brownian motion, it is known that the mean position of all the objects will remain unchanged with time. However, the same is not true for the target object. It is being moved deterministically by the primary trap along a known path with a known velocity. If by the time the plan is executed, it has already moved quite close to an obstacle, then any further motion along a similar direction will definitely result in collision. So, by accounting for this motion, such moves can be avoided and an appropriate circumvention plan can be obtained. All the steps are formally described in the algorithm DYNAMIC_QMDP.

An important issue with the approach is the size of the control action set. It has to be *pruned* in order to obtain results within few ms. The concept of non-dominated control action set proves to be very useful for this purpose. A *non-dominated control action set* is formed by including those control actions such that all of them are non-dominated (not *always* expected to yield an inferior solution) with respect to at least one other control action present in this set. This is formed by selecting those control actions such that the corresponding primary traps are located at the estimated target object center at the instant the plan will be executed and the secondary traps are either switched off or positioned at one of the estimated obstacle centers. Moreover, secondary traps are not switched on if the obstacles are always expected to lie within the collision region or are never expected to drift inside the collision region. The primary trap moves at the maximum speed, unless there is a possibility of collision with the nearest obstacle, which may be prevented by slowing down. Only those primary trap velocity vectors that make angles, which are multiple of 5° with the X-axis, are considered.

Algorithm: DYNAMIC_QMDP
Symbols

- Set of rectangular grid coordinates $X = \{x_1, \ldots, x_N\}$ representing the state space, where N is the total number of grid points
- Value function V
- Belief function b

- Expected payoff function r
- Time parameter t
- Iteration counter q
- Span of a vector $sp(w) = \max_i w_i - \min_i w_i$

Input

- Approximate goal location $x_G \in X$
- Approximate, estimated location of target object center $x_T \in X$
- Set of approximate estimated locations of all the obstacle centers $E = \{y_1, \ldots, y_{n-1}\}$, where $y_i \in X, \forall i$, and n is the total number of objects (including target object) in W
- Set of pruned control action sets $U = \{u_1, \ldots, u_m\}$, where m is the number of control action sets left after pruning
- Set of state transition probability matrices $P = \{p_1, \ldots, p_m\}$, where $p_k \equiv p_k(x_j \mid u_k, x_i) \forall i, j$ represents the matrix corresponding to the k^{th} control action set
- Optical tweezer parameters R_a, Δt_c, t_d, l_{tr}, l_{tw}, σ
- Maximum trap transport velocity vector $v_m = \begin{bmatrix} v_{xm} & v_{ym} \end{bmatrix}^T$; $v_{\max} = |v_m|$
- Previous primary trap velocity vector $v_p = \begin{bmatrix} v_{xp} & v_{yp} \end{bmatrix}^T$

Output

- Optimum control action set u^*

Steps

1. Initialize $t = t_d$ and $q = 0$.
2. **For** $i = 1$ to N, **do** $V^q(x_i, E, t) = \Delta t_c$.
3. **Do while** $q \le 1$ **or** $sp(V^q - V^{q-1}) \ge 1$
   a. **For** $i = 1$ to N, **do**
      i. **For** $k = 1$ to m, **do**
         1. **If** $(x_i == x_T)$, compute the new grid location $x_{n,k} = (x_T + v_p t) + v_k \Delta t_c$, where v_k corresponds to the primary trap velocity vector of control action set u_k; **else** compute $x_{n,k} = x_i + v_k \Delta t_c$
         2. **If** $\|x_i - x_G\| \le R_a$, $r(x_i, E, u_k, t) = 0$; **else if** $\|x_{n,k} - x_G\| \le R_a$, $r(x_i, E, u_k, t) = \Delta t_c$; **else**
$$r(x_i, E, u_k, t) = \min \left\{ \sum_{l=1}^{n_c} p_l t_{avg}, \left(\frac{x_{n,k} x_G}{v_{\max}} \right) \left(\frac{n_s \Delta t_c}{t_e} \right) + \sum_{l=n_s}^{n_c} p_l t_{avg} \right\}$$
(see note at the end of the algorithm for explanation)
$$\hat{V}(x_i, E, u_k, t) = r(x_i, E, u_k, t)$$
         3. Compute
$$+ \sum_{j=1}^{N} V^q(x_j, E, t)\, p(x_j \mid u_k, x_i)$$
         ii. Obtain $V^q(x_i, E, t) = \min_{u_k} \left[\hat{V}(x_i, E, u_k, t) \right]$
      b. Increment q by 1 and t by the actual time taken to complete Step a.
4. **For** $i = 1$ to N, **do**
   a. **For** $k = 1$ to m, **do**
$$Q(x_i, E, u_k, t) = r(x_i, E, u_k, t) + \sum_{j=1}^{N} V^q(x_j, E, t)\, p(x_j \mid u_k, x_i)$$
   b. Sample b_i randomly from a normal distribution with mean and standard deviation equal to x_i and σ respectively and multiply it with the predicted value.
5. Normalize belief function values such that $\sum_{i=1}^{N} b_i = 1$.
6. Return $u^* = \arg\min_{u_k} \sum_{i=1}^{N} b_i Q(x_i, E, u_k, t)$.

It should be noted here that all the objects whose effective obstacle regions intersect the trajectory of the primary trap as it moves from x_i to $x_{n,k}$ and then from $x_{n,k}$ to x_G along a straight line path are considered to be potentially colliding. n_s is the number of activated secondary traps. Trapping probability p_l for the l^{th} potentially colliding object is obtained by considering the minimum distance between its estimated location and the trajectory of the primary trap and utilizing l_{tr} and l_{tw}.

B. Simulated Planning Trajectories

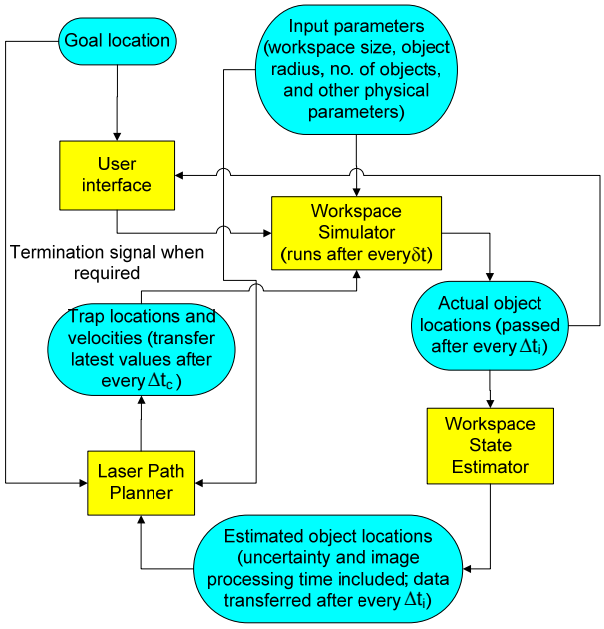


Fig. 2. Simulation framework: Connection between simulator, estimator, path planner, and user interface

We have replicated the overall physical system in form of a simulation framework shown in Fig. 2. The workspace

simulator takes physical parameters as the input and generates the simulated locations of the object centers after every time interval $\delta t < \frac{m}{\gamma}$ [25], which is the relaxation time of the object. Here, m represents the mass of the object. The user interface receives data after every Δt_i (50-500 ms) time interval from the workspace simulator and decides whether the target object has reached the goal location. The workspace state estimator also obtains data at the same rate from the simulator and adds noise to the true locations to simulate imaging system uncertainty. It passes estimated positions of the object centers to the laser path planner with an appropriate delay (t_d always selected as 50 ms) to simulate imaging system latency. The planner then comes up with new locations and velocities of the traps. It transfers the latest values every Δt_c (50-1000 ms) to the workspace simulator.

Three test cases have been presented to visualize sample 2D trajectories generated by the infinite-horizon algorithm. It is implemented in C++, using Microsoft Visual Studio.Net 2003 as the compiler. OpenGL is used for rendering purposes. Here, simulations are performed on 1.0 μm radius glass spheres, immersed in water at 20° C. Correspondingly, l_{tr} and l_{tw} are chosen to be equal to 1.0 μm and 1.5 μm respectively. The dimensions of the workspace are 50 μm X 50 μm. $\Delta t_i, \Delta t_c$ and t_{avg} are all chosen to be 50 ms (applicable for fast scanning mirror tweezer set-up). σ is taken to be 0.25 times the object radius. Laser power and wavelength are selected as 100 mW and 500 nm respectively. Fig. 3 shows the trajectory followed by the target object in a moderately crowded scene consisting of 30 obstacles. The estimated locations of the object centers are shown by '+' signs, whereas, they have been drawn at their actual locations. Brownian motion trajectories of all the untrapped objects are also shown in the Figure. This Figure clearly illustrates the capability of the algorithm to detect the presence of obstacles and follow a circumvented path in order to avoid unwanted collisions.

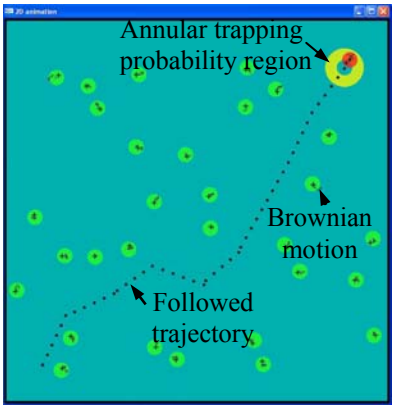
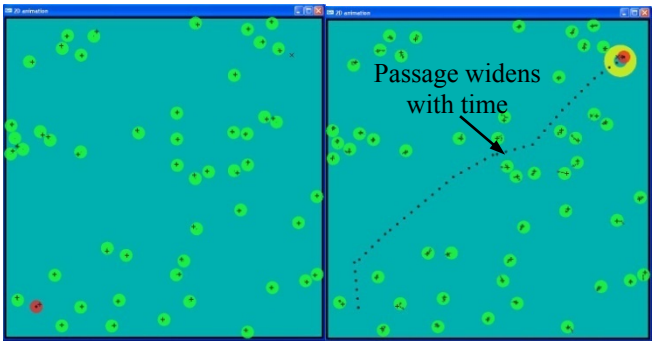


Fig. 3. Circumventing obstacles in a moderately crowded scene

Fig. 4 shows the usefulness of this dynamic programming framework as opposed to a simple, conservative scheme, wherein a search strategy can be employed to just follow the safest route based upon increasing collision regions (due to increase in diffusion length with time) in the *C*-space. Such a conservative scheme will not be able to guide the target object through passages that may have opened up with time. On the other hand, the infinite-horizon algorithm presented here is an adaptive and dynamic approach that generates an optimum solution based upon the new state of the workspace, every time it is invoked. That is why, although the passage was too narrow in the initial workspace state (see Fig. 4(a)), later on it became wider, and the primary trap could successfully transport the target object through this widened passage without causing any unwanted collision(s) (see Fig. 4(b)). This results in an overall decrease of transport time in the range of (25 – 30) % as compared to the conservative scheme.



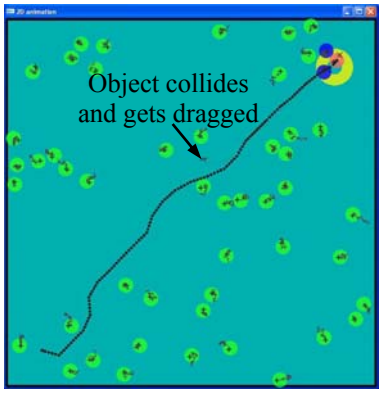
(a) Initial scene (b) Final scene

Fig. 4. Transporting the target object through an originally narrow passage that became wider with time

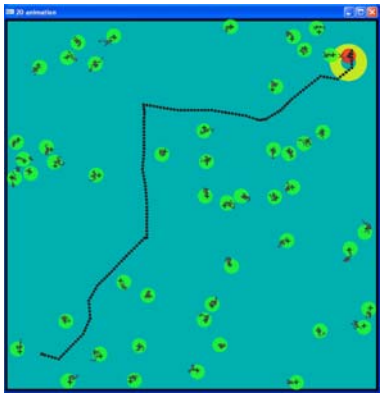
Fig. 5(a) shows another phenomenon in a similarly crowded scene consisting of 50 obstacles. Initial positions of all the objects are identical to that in the last scene. The maximum speed of the primary trap is reduced by 3 times as compared to the value used in the previous two test cases. It can be seen that the effect of Brownian motion is more pronounced in this case, as the diffusion length is proportional to the square root of the elapsed time. Now, during the transport operation, the planner intentionally allows collision with an obstacle. Based upon the current set of payoff function parameters, this turns to be the better option as compared to a circumvented route that involves no collision. Due to small size and momentum of the colliding objects, they do not move away from each other after impact. Instead, they get displaced by such a little amount that both of them remain within the high trapping probability region. Thus, both of them get dragged by the primary trap towards the goal. Another collision event takes place with an obstacle that was located very close to the goal. We have not implemented switching the trap off, and then activating it after the objects have drifted away in the current setting to demonstrate the effect of collisions.

The algorithm provides the option of changing the payoff function parameters so that an alternative non-collision path can also be computed if one wants to avoid trapping multiple objects. This is shown in Fig. 5(b) by increasing the value of

t_{avg} to 150 ms. The initial state is same as that in the previous test case. A much longer, circuitous path is followed by the primary trap that not only avoids the first collision, but also goes around and above the goal before turning back to avoid trapping the obstacle lying close to the goal. This kind of flexibility will be useful in practical applications.



(a) Collision and dragging of obstacle



(b) Avoiding collision and following a longer route

Fig. 5. Trajectories in a crowded scene where the relative rate of change of free *C*-space is high

IV. EXPERIMENTAL VALIDATION

A. Optical Tweezer Instrument and Feedback Control

The optical tweezer used in these experiments was a BioRyx 200 (Arryx, Inc., Chicago, IL) holographic laser tweezer. The BioRyx 200 consists of a Nikon Eclipse TE 200 inverted microscope, a Spectra-Physics Nd-YAG laser (emitting green light of wavelength 532 nm), a spatial light modulator (SLM), and proprietary phase mask generation software running on a desktop PC. Nikon Plan Apo 60x/1.4 NA, DIC H oil-immersion objective is used. In order to impose a trajectory on the beads with a SLM, the optical trap is moved in a series of closely spaced steps aligned along the intended trajectory; the bead follows the steps of the trap quickly and then remains stationary until the next step. The stepping rate and step size determine the speed. The maximum rate at which traps can be set is the update rate of the SLM, 15 Hz, and the minimum step size is 150 nm.

The feedback control was achieved with a second PC equipped with a uEye camera (IDS, Inc., Cambridge, MA) for imaging the beads and software for executing the planning algorithm. A similar optical feedback approach in a system without stochastic fluctuations was implemented previously to control unstable crystal growth patterns [26]. Beads were identified and located by thresholding the image and then calculating the center of mass of all the remaining blobs. For 2.01 μm diameter beads used in the physical planner experiments (see Section IV C), this simple algorithm provided enough accuracy while maintaining a reasonable frame rate (1-2 frames/sec).

B. Sphere Behavior without Optical Traps

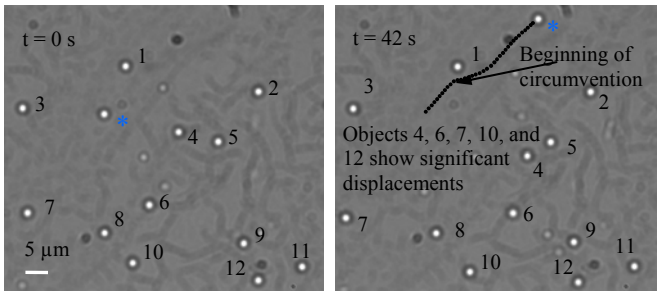
We have observed the behavior of both 1.0 and 2.01 μm nominal diameter amorphous silica spheres in water at room temperature. The spheres have been procured from Bangs Laboratories, Inc., Fishers, IN. Refractive index of the spheres is 1.46 and density is 2000 kg/m^3. In the absence of any optical trap, they freely diffuse in water. Estimation of the average in-plane diffusion length of ten beads each at four imaging planes other than the bottom one between 10 successive frames corresponds well to the theoretically predicted value of 2.16 μm. The same experiment is repeated inside the simulator for the same number of silica spheres (with same material properties) at identical horizontal planes. The values lie within 5% of the experimental numbers. This shows that Brownian motion has been accurately modeled in the simulator.

We have also counted the number of beads lying at the different imaging planes. Reading is continuously taken for about 16 s at every imaging plane; then we move up to the next plane and the same cycle is repeated once more for all the five planes. Four different number values (two each at the beginning and ending of the first and second observations) obtained for every imaging plane are used to fit a linear curve. Values are extrapolated from the fitted curve to obtain the expected number of beads at that particular height at any time instant. We can see a progressive settling of beads at the bottom plate accompanied by a corresponding decrease in the number of beads at all the other planes. A similar experiment has been repeated inside the simulator by placing exactly the same number of spheres at different heights as observed at t = 300 s after the first reading is taken. Simulation is allowed to run for 100 s and then the number of spheres is counted at different z-levels. Values again correspond within 5% of those occurring in the physical set-up. This shows that all the non-trapping forces have been correctly accounted for in the simulation as the number distribution depends on all these forces simultaneously. Physically-obtained number distribution also provides us with a good starting point for all our simulation experiments.

C. Sphere Transportation using Automated Planner

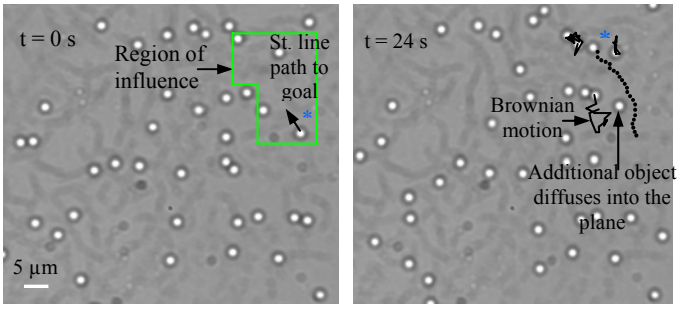
We have been successful in transporting 2.01 μm diameter silica beads using the current planner described in Section IIIA automatically without any manual intervention. The dimension of the camera image is 896 x 800 pixels. An initial and a goal location are specified in the controller user interface by

manually clicking at two spots in the imaged workspace. The bead that is located nearest to the initial location is automatically selected as the target bead. It is then transported to the desired goal location by the POMDP planner; the planner is terminated when the target bead reaches within 1.0 μm radius of the second selected spot. Two such representative experimental runs near the bottom plate are shown in this paper to illustrate the capability of the system. The trap charge is always set to 0.0 to create a point trap.



(a) Initial scene (b) Final scene

Fig. 6. Obstacle circumvention by automated planner in holographic optical tweezer set-up



(a) Initial scene (b) Final scene

Fig. 7. Forced deviation of automated planner from linear trajectory due to obstacle diffusion

Fig. 6 depicts the effective circumvention of an obstacle by the target object (represented by '*' sign by its side) moved by an optical trap at a speed of 2.23 μm/s. Laser power is set at 0.2 W. The time at which each frame is recorded is shown on the top left-hand corner. The overall trajectory of the target object is displayed using dots in Fig. 6(b). A number is assigned to every object so that it can be easily tracked in the two figures. Fig. 7 shows that even if a straight line path exists between the initial position of the target object and the goal, Brownian motion of nearby obstacles and diffusion of out-of-plane objects into the plane under consideration will cause the trap to follow a circuitous route. The random trajectory of the objects that influence the trap motion is also shown in Fig. 7(b). In this experiment, the trap is moved at a faster speed of 4.45 μm/s.

Collision and multiple trapping of beads are found to occur during transportation in crowded scenes after more beads have settled down. When obstacles are present in all possible routes to the goal, the planner decides to move along a relatively shorter path, allowing the possibility of collision with a nearby object. This sometimes results in simultaneous transport of multiple objects to the goal. As in the 3rd simulation test case (Fig. 5(a)), switching off the laser has not been incorporated due to the time it takes for the holographic tweezer to heat up and reach full intensity.

V. SIMULATION RESULTS

We have carried out additional experiments in the validated simulator to investigate the performance of the POMDP algorithm under a wide variety of operating conditions since it is not possible to perform them in the physical set-up due to the time needed to conduct a large number of runs to achieve statistically meaningful results. All the experiments are run on a Pentium 4 processor PC with 3.6 GHz clock speed and 1 GB of RAM. Every experiment is repeated 100 times and the average transport time is used as the performance measure. The workspace size is taken to be 89.6 μm x 80 μm x 50 μm (same as that used for planning experiments). $\Delta t_i, \Delta t_c$ and t_{avg} are always selected as 0.25 s, 0.5 s and 0.5 s respectively so that they are consistent with the values observed in the holographic tweezer set-up. A silica microsphere, having the same properties as used in the physical experiments, is initially positioned at (5 μm, 5 μm) in the XY plane and transported to (85 μm, 75 μm) in the same horizontal plane at room temperature in all the experiments. However, the z-location of the target object is varied in certain cases. Water is always used as the fluid medium. Laser wavelength and numerical aperture of the objective lens are also assigned constant values as specified in Section IV A. Unlike in Section III B, time is not included as a parameter inside the convergence loop of the planning algorithm. This maintains consistency with the fact that the trap in the holographic tweezer stops once it has reached the intended location for a given control action command. However, the algorithm is general enough to deal with other kinds of set-ups where the traps continue moving as long as they are not commanded to act otherwise.

In the first two sets of experiments, we have varied the laser power and the imaging uncertainty standard deviation respectively, keeping all the other factors constant. The first time any experiment is carried out, XY locations of the object centers are assigned following a uniform, random distribution. However, these values are stored and re-used in the other 99 trials. The target object is always placed at the z = 6 μm plane. This implies that the planner initially deals with 50 obstacles due to 2D imaging restrictions. Under such a situation, for 7200 grid points, an initial (non-pruned) primary control action set consisting of 7200 elements at every grid point, and 9 secondary traps that can be activated, average planner computation time is about 60 ms. However, with time the number of obstacles will change as objects diffuse in and out of the horizontal plane. Other parameters remaining constant (as expected in a typical physical set-up), the computation time varies almost linearly between (20-115) ms if the number of obstacles changes from 10 to 100. $v_{\max}$ is chosen for every

data point by carrying out pilot simulation runs before starting the experiments.

The average transport time decreases only slightly from (28.6-26.6) s as the power increases from (0.2-2.0) W at constant σ = 0.125 μm. This happens because $v_{\max}$ increases almost linearly with power; however, l_{tr} and l_{tw} also increase correspondingly meaning that less free *C*-space is available. Thus, although the primary trap can transport the target object faster, it has to circumvent more or activate more number of secondary traps in order to avoid collisions. However, as expected, the average transport time increases significantly from (24-45) s as the imaging uncertainty standard deviation varies from (0.05-1.0) , for a constant value of laser power equal to 0.2 W. For higher values of σ, since the sphere positions will be known with a much lesser degree of certainty, more collisions will take place, thereby increasing the transport time.

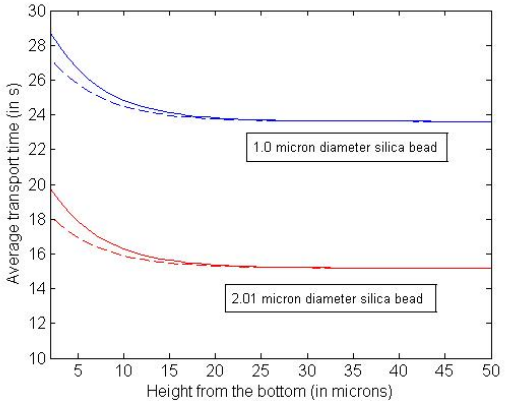


Fig. 8. Planner performance comparison with (dashed lines) and without (solid lines) knowledge about the z-coordinates of the object centers

The last experiment enumerates the extent of performance degradation as a result of ignoring the presence of spheres at other cross-sectional planes. Since they (particularly those lying relatively close by) may also get trapped by any activated laser beam, additional collisions will take place that have not been accounted by the planner. Consequently, the overall transport time will increase. As shown in Fig. 8, this effect can only be seen near the bottom of the plate. This occurs due to the fact that many more objects are present near the bottom plate; hence, a larger number of objects will be undetected and consequently, greater number of unpredicted collisions will happen. The transport time more or less follows an exponential trend in accordance with the distribution of objects in the workspace.

The experiment is performed both for 1.0 and 2.01 μm diameter silica spheres, using laser power of 0.2 W and σ equal to 0.25 times the object radius. $v_{\max}$ is more for the 2.01 μm diameter spheres (7 μm/s) as compared to the 1.0 μm diameter spheres (4.5 μm/s). On the other hand, diffusivity is half for the larger sphere as compared to the smaller one. Together, these two factors counteract the effect of larger horizontal trapping regions for the bigger spheres, thereby resulting in smaller transport times. However, the overall variation trend is similar in both the cases.

It should be noted here that we have not performed the experiments at z = 0 and 1 μm planes because of the fact that additional physical phenomena need to be modeled near the cover slip glass surface. Correction to the viscous drag term [24] (and correspondingly to the diffusivity terms which become anisotropic as well) needs to be made. Moreover, sliding friction and adhesion forces also have to be taken into account in order to transport objects that are stuck to the bottom plate. Another point that should be mentioned here is that in the simulation we have not increased the image processing time when z-locations of the object centers are also determined. But as shown in [27], when data from a stack of images taken at various z-planes are combined together to estimate the z-coordinate, computation time is significant. Thus, the planner may not be that effective as the scene for which the optimal plan is generated will have changed quite a lot by the time the plan is actually implemented. So the planner with 2D imaging may not perform significantly inferior as compared to the one with full 3D imaging.

VI. Conclusion

This paper describes a stochastic dynamic programming framework for transporting a single particle using optical tweezers from its current location to a given goal location in the minimum, expected time. Tweezer systems provide environments where obstacles exhibit random, Brownian motion and both forms of uncertainty (control action and measurement) are inherently present. The main contributions of this work are listed as follows:

- A real-life, physical problem is abstracted and modeled as a path planning problem by developing a stochastic dynamic programming framework.
- An existing, approximate infinite-horizon POMDP algorithm is modified by changing the form of the payoff function and introducing a time parameter inside the convergence loop to enhance computational speed and accuracy.
- Application of the algorithm on test cases clearly shows that it is flexible, works well in crowded scenes, and is able to detect and transport the target object through corridors that were originally too narrow but widened over the course of time.
- Experiments conducted with silica beads provide data to validate the physical accuracy of the simulator. Moreover, successful runs show that the automated planner is capable of transporting a specific object by circumventing obstacles.
- Simulation experiments are carried out to characterize the performance of the planning algorithm under varying conditions of laser power, imaging uncertainty, and object number density.

Directions for future work include extending the basic framework of the infinite-horizon algorithm to perform optimal planning for transport of ensembles of spherical particles, and for transport in three dimensions.

REFERENCES

[1] A. Ashkin, "Forces of a single-beam gradient laser trap on a dielectric sphere in the ray optics regime," *Biophysical J.*, vol. 61, no. 2, pp. 569-582, Feb. 1992.

[2] A. Ashkin, "History of optical trapping and manipulation of small-neutral particle, atoms, and molecules," *IEEE J. Selected Topics in Quantum Electronics*, vol. 6, no. 6, pp. 841-856, Nov./Dec. 2000.

[3] K. Svoboda, and S. Block, "Optical trapping of metallic rayleigh particles," *Optics Letters*, vol. 19, no. 13, pp. 930-932, July 1994.

[4] A. Balijepalli, T. W. LeBrun, and S. K. Gupta, "A flexible system framework for a nanoassembly cell using optical tweezers," in *Proc. ASME International Design Engineering Technical Confs. & Computers and Information in Engineering Conf.*, Philadelphia, PA, Sept. 2006.

[5] J. Curtis, B. Koss, B., and D. Grier, "Dynamic holographic optical tweezers," *Optics Communications,* vol. 207, no. (1-6), pp. 169-175, Jun. 2002.

[6] J. Schwartz, J. Hopcroft, and M. Sharir, Eds. *Planning, Geometry, and Complexity. Robot Motion Aspects*, Norwood, NJ: Ablex Publishing Corporation, 1986.

[7] T. Lozano-Perez, "Spatial planning: a configuration space approach," *IEEE Trans. Computers*, vol. 32, no. 3, pp. 108-120, Feb. 1983.

[8] J. C. Latombe, *Robot Motion Planning*. Boston, MA: Kluwer Academic Publishers, 1991.

[9] V. J. Lumelsky, *Sensing, Intelligence, Motion*. Hoboken, NJ: John Wiley & Sons, Inc., 2006.

[10] E. Masehian, and D. Sedighizadeh, "Classic and Heuristic Approaches in Robot Motion Planning – A Chronological Review," *Proc. World Academy of Science, Engineering and Technology*, vol. 23, pp. 101-106, Aug. 2007.

[11] S. M. LaValle, *Planning Algorithms*. New York, NY: Cambridge University Press, 2006.

[12] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA: The MIT Press, 2005.

[13] S. M. LaValle, and S. A. Hutchinson, "An objective-based framework for motion planning under sensing and control uncertainties," *The International J. Robotics Research*, vol. 17, no. 1, pp. 19-42, Jan. 1998.

[14] S. M. LaValle, "Robot motion planning: A game-theoretic foundation." *Algorithmica*, vol. 26, no. 3, pp. 430-465, 2000.

[15] R. Alterovitz, A. Lim, K. Goldberg, G. S. Chirikjian, and A. M. Okamura, "Steering Flexible Needles under Markov Motion Uncertainty," in *Proc. IEEE/RSJ International Conf. Intelligent Robots and Systems*, Alberta, Canada, 2005, pp. 1570-1575.

[16] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars, " Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces," *IEEE Trans. Robotics and Automation*, vol. 12, no. 4, pp.566-580, Aug. 1996.

[17] A. Sud, E. Andersen, S. Curtis, M. Lin, and D. Manocha, "Real-Time Path Planning for Virtual Agents in Dynamic Environment," in *Proc. 2007 IEEE Virtual Reality Conference*, Charlotte, NC, 2007, pp. 91-98.

[18] S. Prentice, and N. Roy, "The Belief Roadmap: Efficient Planning in Linear POMDPs by Factoring the Covariance", in *Proc. International Symposium on Robotics Research*, Hiroshima, Japan, 2007.

[19] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, L. Kavraki, and S. Thrun., *Principles of Robot Motion: Theory, Algorithms, and Implementation*. Cambridge, MA: The MIT Press, 2006.

[20] J. Bruce, and M. Veloso, "Real-time randomized path planning for robot navigation", in *Proc. IEEE/RSJ International Conf. Intelligent Robots and Systems*, Lausanne, Swizerland, 2002, pp. 2383-2388.

[21] M. Zucker, J. Kuffner, and M. Branicky, "Multipartite RRTs for Rapid Replanning in Dynamic Environments", in *Proc. IEEE International Conf. Robotics and Automation*, Rome, Italy, 2007, 1603-1609.

[22] K. E. Bekris, and L. E. Kavraki, "Greedy but Safe Replanning under Kinodynamic Constraints", in *Proc. IEEE Inrternational Conf. Robotics and Automation*, Rome, Italy, 2007, pp. 704-710.

[23] W. H. Wright, and G. J. Sonek, "Radiation trapping forces on microspheres with optical tweezers," *Applied Physics Letters*, vol. 63, no. 6, pp. 715-717, Aug. 1993.

[24] W. H. Wright, G. J. Sonek, and M. W. Berns, "Parametric study of the forces on microspheres held by optical tweezers," *Applied Optics*, vol. 33, no. 9, pp. 1735-1748, Mar. 1994.

[25] A. G. Banerjee, A. Balijepalli, S. K. Gupta, and T. W. LeBrun, Generating simplified trapping probability models from simulation of optical tweezers system, *J. Computing and Information Science in Engineering*, vol. 9, no. 2, Jun. 2009.

[26] A. J. Pons, A. Karma, S. Akamatsu, M. Newey, A. Pomeranc, H. Singer, and W. Losert, Feedback control of unstable cellular solidification fronts, *Phys. Rev. E*, vol. 75, no. 2, pp. 021602-1-021602-5, Feb. 2007.

[27] T. Peng, A. Balijepalli, S. K. Gupta, and T. LeBrun. Algorithms for on-line monitoring of micro-spheres in an optical tweezers-based assembly cell. *J. Computing and Information Science in Engineering*, vol. 7, no. 4, pp. 330-338, Dec. 2007.