2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

An RRT-Based Path Planner for Use in Trajectory Imitation

Jonathan Claassens

Abstract— **We propose a more robust robot programming by demonstration system planner that produces a reproduction path which satisfies statistical constraints derived from demonstration trajectories and avoids obstacles given the freedom in those constraints. To determine the statistical constraints a Gaussian Mixture Model is fitted to demonstration trajectories. These demonstrations are recorded through kinesthetic teaching of a redundant manipulator. The GMM models the likelihood of configurations given time. The planner is based on Rapidly-exploring Random Tree search with the search tree kept within the statistical model. Collision avoidance is included by not allowing the tree to grow into obstacles. The system is designed to act as a backup for a faster reactive planner that may fall into a local minimum.**

To illustrate its performance an experiment is conducted where the system is taught to open a Pelican case using a Barrett Whole Arm Manipulator (WAM). During reproduction an obstacle is placed nearby the case to partially obstruct the manipulator. The planner successfully avoided this obstacle without drifting from the trend in the demonstrations.

I. INTRODUCTION

The field of robot programming by demonstration, or imitation, is concerned with the efficient transfer of behaviour from a human demonstrator to an observing robot. This approach promises to greatly minimize the amount of expert domain knowledge a user needs to operate a robot. It also provides a more natural interface to the system and should reduce programming effort. For an overview of the field see [1].

Imitation can be conducted at a number of levels of abstraction. In higher level approaches [2] [3] the focus is organization of behaviour primitives. This paper addresses lower, trajectory level imitation where the concern is on how to reproduce a behaviour trajectory given a set of demonstration recordings. The approach assumes that the user is trying to demonstrate a behaviour as consistently as possible. Large variation in some portion of the demonstrations is assumed to suggest some liberty when performing the task.

In much of the literature on trajectory imitation [4] [5] [6] [7], the focus has been on producing a statistically likely or generally successful path where collision avoidance is largely ignored. Our application, a robotic lab technician, requires a different emphasis. With a constant threat of collision and a broad variety of different situations in which imitation must be conducted we require a robust scheme. This is especially true in a production line where a manufacturer cannot afford to have a robot get 'stuck'. A robust scheme would require a search for some reproduction trajectory that not only satisfies the statistical constraints determined through some demonstration characterization phase, but also avoids obstacles and joint limits. Obstacles can, of course, contact any part of the robot.

Path planning, where the emphasis is on finding some safe path from a start to a destination, has often been conducted using the Rapidly exploring Random Tree (RRT) concept [8]. The planner's requirements for statistical completeness are very undemanding. This planner was chosen for adaptation into the present work over alternative algorithms such as the probabilistic roadmap or randomized potential fields planners because it is straight-forward to include complicated manipulator dynamics and collision information into the algorithm. Plant dynamics are not considered in this paper, but will be added in future efforts.

The RRT algorithm can be explained briefly as follows. A tree is initialized with the root set at the required start position. Subgoals are randomly chosen uniformly across the search space. For each subgoal, the nearest tree node is determined and a child node is instantiated for it by moving (through simulation of the plant model) from the selected node a short distance toward the subgoal. If a collision occurs enroute then the growth is halted and the child is created just short of collision. When a node approaches the goal and a path exists from it to the goal, the path from this node to the tree root is a solution. Because the method is statistically complete it will find a path to a goal assuming one exists.

This paper proposes an RRT-based planning algorithm which attempts to find some configuration space path that avoids collision and imitates a behaviour by remaining within statistically determined constraints. It can be used in parallel with a faster, reactive planner as a backup measure in case the reactive planner is unable to find a solution. The platform chosen to illustrate these ideas is a redundant manipulator, a Barrett WAM, with seven degrees of freedom (DOF). The additional free DOF can be exploited to find some collision free path. Optimization of the path with respect to some time or smoothness metric is left for future work.

Demonstration data is collected through kinesthetic teaching as in [4]. The WAM simply compensates for gravity and allows the user to move it through some motion. The grasper, a Barrett Hand, can be adjusted in the same way at any point. Alternative data collection schemes used in related works include recording demonstrator posture using markerless colour tracking algorithms [9] or marker based vision schemes [7]. The kinesthetic approach was chosen because as the user illustrates a motion to the robot he/she is made

The research was supported by funding from the Council for Scientific and Industrial Research (CSIR), South Africa.

J. Claassens is a research engineer with the Mobile Intelligent Autonomous Systems (MIAS) group in the Council of Scientific and Industrial Research (CSIR), South Africa, jclaassens@csir.co.za

fully aware of the robot's mechanical limitations. This also completely circumvents the correspondance problem which is not a focus of the work.

An imitation path planner requires some statistical model of a taught behaviour. The statistical constraint used in the work closely resembles the method discussed in [4] [10]. A Gaussian mixture model (GMM) is fitted over a set of demonstration trajectories using the expectation maximization algorithm. The GMM is used to model the likelihood of a trajectory passing through a portion of space at a certain time when the behaviour is being executed. A brief description will be given in section 4, but for an indepth understanding the reader should consult [10].

The structure of the paper is as follows. Section 2 will discuss related work and Section 3 will present the data collection method. Section 4 will briefly describe the approach used to statistically characterize the demonstrations trajectory. The proposed planner is described in detail in Section 5 with experimental results illustrating its performance in Section 6. Section 7 gives conclusions and future work.

II. RELATED WORK

A work with a similar goal is described in [9]. In the work, statistical characterization is conducted in a manner similar to the approach in [4], but reproduction is performed with collision avoidance in mind. Their proposed reproduction method relies on gradient descent (or Levenberg-Marquardt) to optimize a path to minimize a cost function which includes a collision term. It is based on previous work in [11]. The approach is designed to be fast. Our emphasis is different. Because such methods cannot eventually guarantee a path if one is possible, in other words they are not statistically complete, we instead chose to use a robust general planner based on the RRT scheme.

The Dynamic Movement Primitive (DMP) [6] based system described in [12] adjusts parameters of a controlling differential equation to avoid obstacles whilst mimicking the behaviour of a tutor. It is also essentially reactive in its collision avoidance and no general search is conducted to guarantee a feasible path. Their priority was also speed.

The imitation scheme in [7] attempts to detect collision during simulation of the reproduction path. If detected, the system simply halts. In [13], a method for planning the grasping of an object is presented which uses models extracted from demonstrations. The work applies to grasping and not to general interaction situations like opening a case.

Outside imitation, planning with manifold or volume constaints has been addressed using an RRT [14]. Their planner requires a manifold constraint to be specified beforehand so that the RRT variant can bridge start and goal positions through the obstacle.

III. DATA COLLECTION

As discussed, demonstrations are recorded through kinesthic teaching. A user is asked to demonstrate the task as accurately as necessary a number of times. The end-effector position and orientation are regularly logged with a timestamp after contact is made with the environment. Thus a trajectory point is a 7 dimensional vector with the first value being time, the next three being position and the last three, orientation in spin-axis form (the axis of rotation times angle of rotation required to transform the base frame to the end-effector frame). The elbow position is ignored. For the work presented in the paper, the manipulator interacts with a single object. The frame of this single object is used as the base frame in which to process all the recordings. This is done because that frame is the one of importance to the user when he/she manipulates the object.

To localize the object, small retroreflectors are attached randomly on its surface. A Riegl 3D imaging laser scanner is used to produce a point cloud of the robot's immediate vicinity. The points' return intensities are thresholded for particularly high values and a clustering algorithm is applied to the result to identify retroreflector centers and orientation. The Iterated Closest Point (ICP) algorithm [15] is used to align two recordings' retroreflector centers to bring the trajectories into a common frame.

IV. DEMONSTRATION STATISTICAL CHARACTERIZATION

To ensure that the trajectories are in temporal agreement, ie. that similar behavioural nuances occur at the same time, we apply Dynamic Time Warping (DTW) [16] as in [9]. The rest of the characterization approach used in the proposed system is an approximation of the approach discussed in [4].

All points from the set of warped demonstration trajectories are modelled as sample points independently drawn from a Gaussian Mixture Model (GMM) given by

$$p(\bar{x}) = \sum_{k=1}^{K} \pi_k n(\bar{x}|\bar{\mu}_k, \Sigma_k) \tag{1}$$

where K is the number of components and $n(\bar{x}|\bar{\mu}_k, \Sigma_k)$ is the normal density with covariance matrix Σ_k and mean $\bar{\mu}_k$. The GMM parameters are learnt through the use of the Expectation-Maximization (EM) [17] algorithm. The EM algorithm is seeded with an initial estimate of density centers calculated with the k-means algorithm. This method seeks to wrap the data in a statistically defined corridor which captures the allowances available during reproduction. Trajectory segments of low required accuracy have mixture components of large variance and high accuracy segments have low variance.

To determine the number of mixture components to use, the Bayesian Information Criterion (BIC) is employed. To do this, a number of mixture models are estimated with different numbers of components. The range used is from 1 to 30 components. The estimation is calculated roughly using k-means. For an explanation of the advantages of this approach see [9]. The BIC metric is calculated for every model using

$$BIC = -2L + P\ln(N) \tag{2}$$

where L is the likelihood of the data given by

$$L = \sum_{i=1}^{N} \log \big(\sum_{k=1}^{K} \pi_k n(\bar{x}|\bar{\mu}_k, \Sigma_k) \big) \tag{3}$$

and P is

$$P = (K-1) + K\big(D + \frac{1}{2}D(D+1)\big). \tag{4}$$

N is the number of sample points and D is the dimension of the data. The model with the lowest BIC is used.

A typical result is a set of densities strung together that run along the mean of the trajectories. Generally, patches of low point probability can appear between densities along the mean curve. For the reproduction planner discussed later we require sequential mixture components' regions of higher likelihood to overlap along the mean trajectory so that no 'gaps' exist. This is because the planner uses a potential reproduction path point's likelihood in the mixture model to decide whether or not to ignore it. Thus the 'gaps' would present artificial boundaries. Another problem is that, occasionally, multiple parallel (with overlapping time) mixture components appear.

To correct these issues an approximation of the result is created. The mixture components are sorted by mean time. All the other dimensions of each are marginalized away to leave just densities on time. For the planner presented in this paper an actual GMM is not required. All that is necessary is a sequential set of Gaussians that envelope sections of space and time to form a statistical corridor. The marginalized result of the components can be used to determine when, in time, that component is dominant. The equation

$$t_{mid} = \frac{\mu_{t,i}\sigma_{t,i+1} + \mu_{t,i+1}\sigma_{t,i}}{\sigma_{t,i} + \sigma_{t,i+1}} \tag{5}$$

can be used to determine the transistion point between which two sequential mixture components switch dominance. It is the average of the means weighted by standard deviation.

These transistions define a series of intervals which can be used to segment the warped trajectories into sets. Reestimating the densities by using the elements of a component's set plus some overlap of the previous and future sets produces the series of Gaussian densities used by the planner. These densities are 6 DOF with time left out because the dynamics of a behaviour are not considered in this paper. The process is illustrated in Fig. 1 and Fig. 2 on 2D synthetic data. An overlap of 0.05 was used on either side of each interval to produce the example. Note that in Fig. 2 that the multiple modes have been flattened in two sequential densities. The mixture components also overlap more than in the EM case.

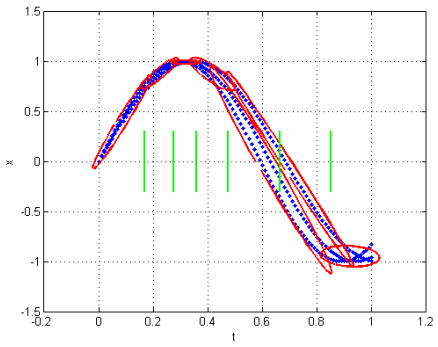

Fig. 1. A normal EM calculated GMM over a set of trajectories. Red ellipses represent 2x standard deviation areas of the mixture components. The data is in blue. The green bars represent intervals calculated using Equation 5.

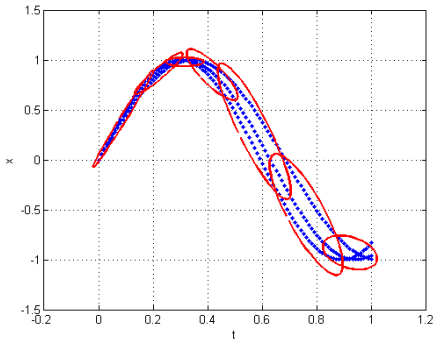

Fig. 2. The approximated string of Gaussians produced as a result of characterization. Red ellipses represent 2x standard deviation areas of the components.

V. THE REPRODUCTION PLANNER

A. The RRT

Algorithm 1 shows the reproduction RRT process. In the code, N represents the model and the symbol $N(i)$ represents mixture component i. The other symbols will be defined throughout the explanation. Initially a tree T is initialized with a root node set at some starting configuration. A standard RRT chooses random subgoals around some fixed goal point. In the proposed algorithm, the subgoals are chosen around the mixture components of the GMM with the statistical variance of those mixture components. This approach will not upset the statistical completeness of the RRT when taking into consideration the statistical constraints. To see why this is the case consider the portion of space outside the statistical constraints to be a single obstacle. Subgoals are still chosen randomly within the unobstructed space, but will lead in front of the tree.

The variable c is the current sample mixture component number. After one iteration of the algorithm it is incremented unless it is equal to the number of the latest mixture component contacted by the tree (m) in which case it is set to zero (the number of the first component).

The random point chosen, $\bar{x}_{rand}$, is in the form of a 6 DOF vector consisting of position and orientation of the end-effector. If the random point is outside two standard deviations of the mixture component then it is discarded and another is selected. This is to prevent highly unlikely points from pulling the planner away from a reasonable result. If the planner has to operate more conservatively, then this check can be made more intolerant. A random inverse kinematic (IK) solution, $\bar{q}_{rand}$, is chosen for the sample. This

solution must not collide with the environment and must not violate any manipulator joint constraints. The function 'ValidConfiguration' should check this. If there is no valid IK solution another random subgoal is selected.

Algorithm 1 ReproductionRRT($\bar{q}_{start}$, $\bar{q}_{end}$, N)

```
1:  T ← (q̄_start, 0)
2:  m ← 1
3:  c ← 0
4:  while TimeRemaining() do
5:    x̄_rand ← RandomPoint(N(c))
6:    if Within2SD(x̄_rand, N(c)) then
7:      q̄_rand ← RandomIKSolution(x̄_rand)
8:      if ValidConfiguration(q̄_rand) then
9:        q̄_nearest ← NearestNeighbor(T, q̄_rand, c)
10:       q̄_step ← Extend(q̄_nearest, q̄_rand)
11:       x̄_step ← ForwardKinematics(q̄_step), N(c+1)
12:       if Within2SD(x̄_step) then
13:         T.AddNode(q̄_step, c+1)
14:         if c+1 > m then
15:           m ← c+1
16:           if m = f then
17:             P ← ExtractPathFromChild(q̄_step)
18:             return P
19:           end if
20:         end if
21:       else
22:         T.AddNode(q̄_step, c)
23:       end if
24:       c ← c+1
25:       if c ≥ m then
26:         c ← 0
27:       end if
28:     end if
29:   end if
30: end while
31: return failed
```

The standard RRT algorithm then searches for the closest node in the tree from which to grow. The proposed algorithm differs in that it selects the closest node which has met a number of criteria (this is done in the 'NearestNeighbour' function). The first is a temporal criterion. Recall that the mixture components are sorted in time. We require a node to be pulled to a mixture component k only if it has entered all mixture components up to and including k. This is the first criterion. It is enforced by associating every node with a score equal to the number of the latest mixture model it has visited. The root (line 1) has a score of 0.

The second criterion is that the node is within two standard deviations of its current, associated mixture component (the latest mixture component through which it has passed) or the subgoal component. Again, as with the check on how unlikely a random sample ($\bar{x}_{rand}$) is, the planner can be made more conservative by adjusting this criterion so that a point must be within, say, one standard deviation.

As in the standard RRT a new node is added to the tree by growing the selected node a short distance toward the subgoal configuration (done in the 'Extend' function). To move the manipulator a short distance toward q_{rand} a direction vector is first constructed using

$$q_{dir} = q_{step} - q_{rand}/|q_{step} - q_{rand}| \tag{6}$$

. This vector multiplied by a small magnitude is added to q_{step} to produce the new node configuration. There are alternative interpolation schemes. An example is calculating the Jacobian transpose or pseudoinverse and using them to determine q_{dir} so that motion of the end-effector is approximately directly toward the subgoal. The simpler method of Equation 6 produces satisfactory results without much computation penalty. To test for collision with the environment, configurations are sampled regularly along the interpolated route and tested using some mesh-to-mesh collision detection package.

If this growth enters the current mixture component (labelled c) then the node is associated with that component (tested on line 11). This node is then allowed to grow into the next mixture component. If this next mixture component number is greater than m, m is incremented.

Having m grow as mixture component waypoints are reached prevents subgoals from being selected from mixture components that cannot currently be reached. The algorithm stopping condition is when the tree reaches the neighbourhood of the final mixture component (labelled f) or when the maximum allowed search time is exceeded.

B. Path Refinement

The configuration space path generated by the reproduction RRT is rough and may contain substantial redundant backstepping. To refine the path for a more visually appealling, faster (fewer waypoints) solution Algorithm 2 is used. The algorithm removes redundant waypoints in the plan. P is the path provide by Algorithm 1 and M is the GMM.

The variable C is used to ensure that the algorithm runs until there are no changes to the path. Function 'ValidMotion' interpolates between the path nodes provided to it using the principle in Equation 6 and checks for collisions. The following if statement (line 10) checks that the average likelihood (calculated in function 'AvgLikelihood') of the path that skips node $i+1$ is not lower than the original section of the path. This likelihood is, of course, determined using the model, M, discussed in Section 4.

The path refinement algorithm runs through the plan and checks whether any step between two points can be removed without having the manipulator collide or reducing the path overall likelihood. If so, this point is removed.

VI. EXPERIMENT

The experimental setup is show in Fig. 4. A Barrett WAM was clamped to an office bench beside a Pelican camera case. The task of the experiment is to teach the system to open the case. Two meters from the Barrett a Riegl 3D imaging laser scanner was bolted into a tripod and used to capture

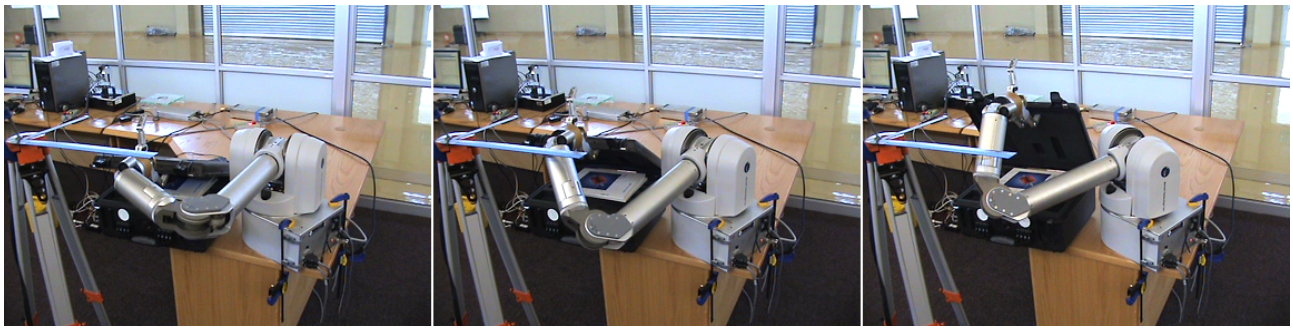

Fig. 3. The execution of a plan which involves avoiding a close obstacle.

Algorithm 2 PathRefine(P, M)

$C \leftarrow 1$
while $C > 0$ **do**
  $C \leftarrow 0$
  $i \leftarrow 1$
  while ($i \leq$ Length(P) -2) and ($C = 0$) **do**
    if ValidMotion($P(i)$, $P(i+2)$) **then**
      L_1 = AvgLikelihood(Motion($P(i)$, $P(i+2)$, M))
      L_2 = AvgLikelihood(Motion($P(i)$, $P(i+1)$, M))
      L_3 = AvgLikelihood(Motion($P(i+1)$, $P(i+2)$, M))
      if $L_1 \geq \frac{(L_2+L_3)}{2}$ **then**
        RemoveFromPath(P, $i+1$)
        $C \leftarrow 1$
      end if
    end if
    $i \leftarrow i+1$
  end while
end while
return P

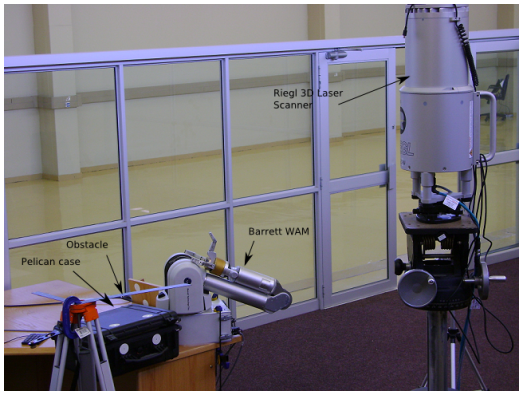

Fig. 4. The experimental setup.

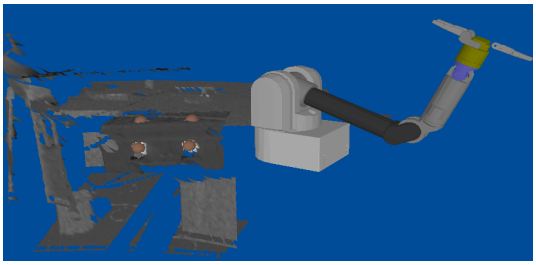

Fig. 5. A screenshot of the simulator illustrating the meshes used and typical results from the laser scanner.

an environment mesh around the case. Its position relative to the Barrett was carefully hand calibrated. As mentioned earlier, the scanner point cloud was also used to determine the pose of the case. The retroreflectors used as fiducials are the white discs pasted to the front of the case.

An obstacle, a flat bar, is put in front of the case to obstruct the Barrett during reproduction. When demonstrations are recorded the obstacle is removed. The attached video shows the recordings and a reproduction solution (at 2x speed). Planning is conducted in a in-house simulation package which is similar to OpenRAVE [18]. The point cloud of the environment provided by the scanner is used to produce a mesh. A mesh of the WAM for a given set of joint angles was generated using data from OpenRAVE. A rendition of the meshes and laser scanner results used is shown in Fig. 5. The orange bulbs on the environment mesh represent retroreflectors on the case. To determine collision between these two meshes during planning the package OPCODE [19] was used.

To find IK solutions for the Barrett there are a number of approaches. Slower methods include Jacobian-transpose and pseudo-inverse iteration, but for planning where this operation has to be performed many times a fast pseudo-analytical method is necessary. The program IKFast that is provided with OpenRAVE provides this.

The plan was produced in two stages. The reproduction RRT is used to produce a solution from a relatively open portion of space just away from the case front. A second basic RRT scheme is used to generate the path from the manipulator's home position to the first configuration of the reproduction path.

Fig. 3 shows a few frames from the produced plan. In the second frame the manipulator can be seen shifting under the bar with the elbow trailing safely behind under the obstacle. The hand will not turn upward and drop the wrist to make more room because this was never demonstrated to the system. It will only operate within the envelope of the

demonstrations.

It should be noted that the gap, under the lid through which the fingers must slide to open the case, is small. When the task was demonstrated the user was more precise in this portion and so the statistical constraints will allow little variance when planning there.

Fig. 6 shows the statistical constraints and the recorded trajectories on common axes. The blue bulbs are the mixture components' surfaces of two standard deviations plotted over each other with some transparency. This allows one to see the resulting corridor through which the planner must traverse. To obtain a 3D density, which is necessary for plotting, from the 6D density which is produced by the characterization stage, the last three orientation variables are marginalized away. Fig. 7 shows the same statistical constraints with the planned path.

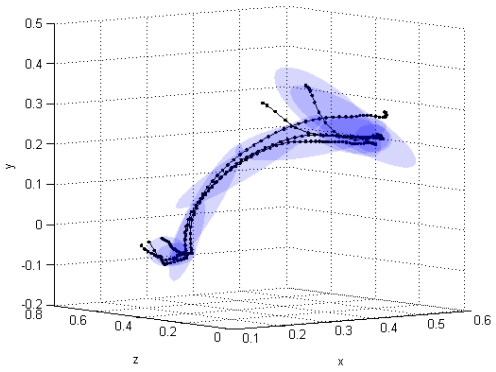

Fig. 6. The statistical constraints (blue band) and recorded trajectories (black connected dots).

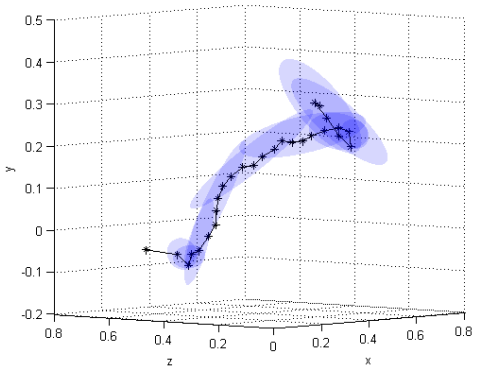

Fig. 7. The statistical constraints and the planned end-effector path.

VII. CONCLUSIONS AND FUTURE WORK

A planning algorithm was presented which is capable of satisfying statistical (imitation) constraints, extracted from demonstrations, and collision avoidance constraints simultaneously. The scheme was successfully demonstrated on a Barrett WAM with the objective of opening a case whilst avoiding a close obstacle.

In future work, the statistical constraints will be replaced with a model which captures smoothness, velocity and more precise directional information so that the planner can tackle dynamic problems. The current run-time of the algorithm is in the order of five minutes on a Pentium Dual-Core 1.80Ghz PC. Optimization to bring this within a few seconds will be a major focus of future work.

VIII. ACKNOWLEDGEMENTS

The author would like to thank Simukai Utete, Deon Sabatta and Giresh Singh for their suggestions.

REFERENCES

[1] B. Siciliano, and O. Khatib (Eds.), Springer Handbook of Robotics, Springer, 2008.

[2] M. N. Nicolescu and M. J. Mataric, Task learning through imitation and human-robot interaction, In K. Dautenhahn and C. Nehaniv, editors, Imitation and social learning in robots, humans and animals: behavioral, social and communicative dimensions, Cambridge University Press, 2005.

[3] M. Pardowitz, B. Glaser, R. Dillman, Learning repetitive robot programs from demonstrations using version space algebra, *proceedings of Robotics and Applications and Telematics*, 2007.

[4] S. Calinon, A. Billard, A Probabilistic Programming by Demonstration Framework Handling Constraints in Joint Space and Task Space, *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, 2008.

[5] D. Lee, N. Nakamura, Mimesis Scheme using a Monocular Vision System on a Humanoid Robot, *In Proceedings of IEEE Int. Conference on Robotics and Automation*, pp. 2162-2168, 2007.

[6] S. Schaal, J. Peters, J. Nakanishi, and A. Ijspeert, Learning Movement Primitives, *International Symposium on Robotics Research (ISRR 2003)*, Springer Tracts in Advanced Robotics, Ciena, Italy: Springer, 2004.

[7] A. P. Shon, J. J. Storz, R. P. N. Rao, Towards a Real-Time Bayesian Imitation System for a Humanoid Robot, *IEEE Intl Conf. on Robotics and Automation (ICRA)*, 2007.

[8] S. M. Lavalle, J. J. Kuffner, Rapidly-Exploring Random Trees: Progress and Prospects, *Jr. of Algorithmic and Computational Robotics: New Directions*, 2000.

[9] M. Muehlig, M. Gienger, S. Hellbach, J. J. Steil, C. Goerick, Task-level Imitation Learning using Variance-based Movement Optimization, *IEEE Intl Conf. on Robotics and Automation (ICRA)*, 2009.

[10] S. Calinon, F. Guenter, A. Billard, On Learning, Representing and Generalizing a Task in a Humanoid Robot, *IEEE Trans. on Systems, Man and Cybernetics*, Part B, vol. 37, no. 2, pp. 286-298, 2007.

[11] M. Toussaint, M. Gienger, C. Goerick, Optimization of Sequential Attractor-Based Movement for Compact Behaviour Generation, *Proceedings of the IEEE-RAS/RSJ International Conference on Humanoid Robots*, 2007.

[12] P. Pastor, H. Hoffmann, T. Asfour, S. Schaal, Learning and Generalization of Motor Skills by Learning from Demonstration, *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 2009.

[13] K. Hsiao and T. Lozano-Perez, Imitation Learning of Whole-Body Grasps, *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, 2006.

[14] D. Berenson, S. S. Srinivasa, D. Ferguson, J. J. Kuffner, Manipulation Planning on Constraint Manifolds, *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 2009.

[15] P. J. Besl, N. D. McKay, A Method for Registration of 3-D Shapes, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239-256, 1992.

[16] C. A. Ratanamahatana, E. Keogh, Everything You Know about Dynamic Time Warping is Wrong, *Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004.

[17] D. J. C. MacKay, Information Theory, Inference, and Learning Algorithms, Cambridge University Press, 2003.

[18] R. Diankov, J. Kuffner, OpenRAVE: A Planning Architecture for Autonomous Robotics, tech. report CMU-RI-TR-08-34, Robotics Institute, Carnegie Mellon University, July, 2008.

[19] P. Terdiman, www.codercorner.com/Opcode.htm, last accessed on the 7th of September 2009.