A Statistical Imitation System using Relational Interest Points and Gaussian Mixture Models

Jonathan Claassens

Council of Scientific and Industrial Research (CSIR)
Meiring Naude Road, Pretoria, 00001
jclaassens@csir.co.za

Abstract

An imitation system that promises applicability to a wide range of problems and platforms is presented. The system's structure is inspired by the Robot Programming by Demonstration (RPD) paradigm. It statistically characterizes demonstration trajectories in the task or joint space. Two sets of information are captured. The first, relational interest points, which are milestone locations taken relative to scene objects, are used to capture important behavioural requirements. The second set of information comes from between these interest points, where the reasons behind a behaviour are less clear, a Gaussian mixture model is used to statistically model the subtrajectories.

In behaviour reproduction nonlinear optimization is used to produce a path of highest likelihood. This approach allows other influences such as collision avoidance to be incorporated into the planner. The statistical model can also be used in behaviour recognition. A simple drawing experiment is used to demonstrate the proposed system and its performance.

1. INTRODUCTION

Suppose a designer needs to implement behaviour in a robotic platform. He or she may do this through hard coding or attempt to use machine learning techniques. These two options are extremes on either end of a spectrum. Hard coded, deterministic methods are typically not adaptive and can be time consuming to implement. Code maintenance may also be prohibitively costly. Pure machine learning techniques such as reinforcement learning are adaptive and do not require extensive user intervention, but suffer from requiring many trial iterations and often obtain poor convergence. Imitation based behaviour sits in the middle of this spectrum. This field of research, also called Robot Programming by Demonstration (RPD), is receiving increasing attention [1]. Of the well documented advantages this approach has to offer, the author considers the most important being that an ideal imitation system will do exactly what the user (or teacher) "expects". This is especially true in industrial applications where emergent behaviour could disrupt a process.

In the past, many complete imitation systems have been proposed. A number require a user to specify behavioural primitives in hard code and the systems use these to segment demonstration traces [2] [3]. For specific applications this approach does have an important advantage in the ability to optimize portions of the behaviour for better accuracy or higher speed. It may, however, filter away important nuances in training data and generally requires a fair amount of user intervention. For more general imitation capability statistical approaches have been applied. Amongst these are Hidden Markov Model (HMM) [4] and Gaussian Mixture Model (GMM) [5] [6] based approaches. The former tends to suffer from producing discontinuities during reproduction from the statistical model. It is, nonetheless, finding application in the imitation of human motion captured from a monocular camera. The GMM approach has been used with Gaussian Mixture Regression (GMR) to produce smooth reproductions of a number of fairly complex behaviours including food preparation and playing pong with a joystick.

This paper proposes an imitation system which characterizes a particular behaviour into a statistical model, reproduces the behaviour smoothly and can recognize recurrence of that behaviour. It relies on a similar GMM approach to the above cited work [5], but differs in how reproduction is conducted and in how relational milestones are specified. The system is designed to require as little user intervention as possible with the exception of a few demonstrations of a particular behaviour. The system makes use of a relational interest point (RIP) detector to identify what locations in the task/joint space are important. The idea of identifying relational milestones in a demonstration so that reproduction can be faithful to a user's intention is an old one [7]. In the proposed system these milestones are defined with a fair amount of generality to capture a broad range of intentions. They are treated as hypotheses that can be rejected as newer trajectories are added to the dataset.

Traces or subtrajectories between RIPs are then characterized statistically using Gaussian Mixture Models. A powerful variant of the Expectation Maximization algorithm is used to determine the model parameters. Unlike [5], which also uses GMMs, the proposed system does not use GMR to reproduce subtrajectories between RIPs. GMR will typically provide some form of mean path, however a mean path may not be highly probable in the density. Instead, nonlinear optimization is used to produce a path of high likelihood. A variant of Nelder-Mead is applied where the cost function is set up so that unlikely points are expensive. There are a number of advantages to this approach. Firstly, the algorithm can be modified to perform collision avoidance much like elastic path planning algorithms [8]. Collision avoidance, in the form of something like a potential field, would be another cost term. Secondly, other characteristics of paths may be tailored such as minimum radius of curvature.

The paper presents the entire system and thus will not focus too heavily on any one component. Its structure is as follows. Firstly, the idea of relational interest points is discussed in the next subsection. Section 2 elaborates on the imitation system architecture by looking at its two major components separately: characterization and reproduction. Experimental results are used to illustrate explanations in Section 2, but are also discussed in detail in Section 3. Section 4 presents the conclusions

and future work.

1.1. Relational Interest Points

To copy a task, one must identify locations relative to surrounding objects that are consistently reached or acted from. An example could be lifting a hot coffee cup. A person will tend to lift the cup by the handle. He or she will do this consistently. A learner who is trying to decifer what is important in such a demonstration, with task reproduction in mind, should note this strategic lifting point.

These characterizations would also need to be defined as random variables because if they are used to describe a set of task demonstrations then they must make allowances for low precision. The term given to these location descriptions in their statistical sense is Relational Interest Point (RIP).

It is assumed that there are many possible types of RIPs that may influence the demonstrator's behaviours. A more complex example shown in Fig. 1 could be characterized with

$$\bar{x}_{justbehind} = (\bar{x}_{ref1} - \bar{x}_{ref2})(1 + K) + \bar{x}_{ref2} \qquad (1)$$

where the bar denotes a vector quantity, $\bar{x}_{ref1}$ is the position of the green circle, $\bar{x}_{ref2}$ the blue.

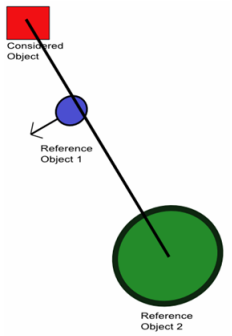

Figure 1: A relative RIP

2. SYSTEM ARCHITECTURE

The system architecture can be divided into two major components: the behaviour characterization component and the reproduction component. The first produces a statistical model of a behaviour given a number of demonstrations. The model's statistical nature allows it to be used in recognition of a behaviour. Reproduction relies on this model and additional environmental observations to reproduce the behaviour. This section discusses each of these two components separately.

2.1. Behaviour Characterization

The explanation of the characterization process will be by means of an example. The experimental setup used is the following. A 6 DOF (Degree of Freedom) Phantom Premium haptics input device was used as both the recording device and as the reproduction platform. During reproduction the Phantom was controlled with a PD (Proportional-Derivative) set point controller that was fed points provided by imitation architecture. A paint brush was attached to the mid limb to enable the setup to draw. A photo of the setup is shown in Fig. 4. For simplicity only the cartesian location of the paint brush nib was recorded. The algorithms provided in the paper can handle more degrees of freedom. Each measurement was stored with a timestamp.

The example is a line drawing exercise. In the demonstrated behaviour, the paint brush is moved into a ink cup, then toward a sheet of paper and finally used to draw a line. Fig. 2 shows the three recorded trajectories.

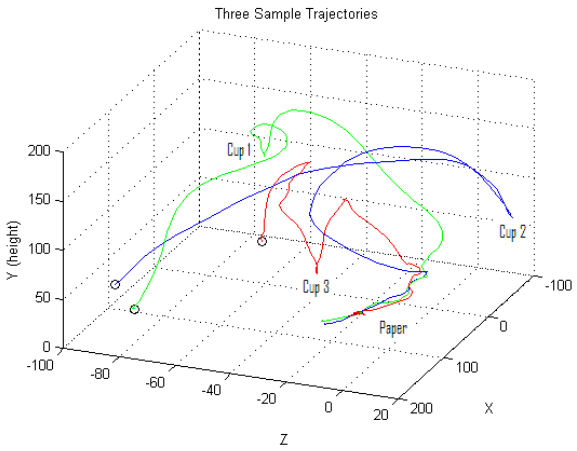

Figure 2: Three sample trajectories with each in a different colour. The black circles indicate the start of a recording. In the demonstration the nib was taken to an ink cup and then a line was draw on the paper.

The input requirements of the characterization process are the recorded trajectories as well as the coordinates of all relevant objects. By "relevant", "necessary for proper reproduction" is meant. The name used to refer to these coordinates is "fiducial" (a term often used in NC machine literature). In the example case, these would be: the 'ink cup', the 'paper' and the start position. The characterization process proceeds as follows.

2.1.1. RIP extraction

In the system, RIPs are considered characteristics of a behaviour that must be reproduced faithfully. The statistical models (density functions) used to characterize subtrajectories between RIPs are rough approximations of what has been seen before so that even in the event of misunderstanding some useful behaviour may be obtained.

The simplest RIP used is relative displacement

$$\bar{x}_{rip} = \bar{x}_0 + \bar{x}_a. \qquad (2)$$

The $\bar{x}_a$ is a random variable vector. To determine if it exists in a set of demonstrations the following algorithm is used:

(1) For every fiducial j do the following:

(1.1) Subtract the fiducial position from the manipulator's position in each of the demonstration trajectories. The resulting quantities are

$$\bar{x}_{rel,j}(t) = \bar{x}_m(t) - \bar{x}_{fid,j}(t) \qquad (3)$$

where $\bar{x}_m(t)$ is the manipulator's position with time and $\bar{x}_{fid,j}$ is the j'th fiducial's location.

(1.2) For every possible permutation $(T_1, T_2, ..., T_N)$, where T_i represents a sample time of trajectory i, do the following:

Table 1: An Example Raw Set of Relational Interest Points

RIP No.	Time in traj. 1.	Time in traj. 2.	Time in traj. 3.
1	223	221	158
2	248	186	123
3	315	98	109
4	319	90	103
5	457	401	252

(1.2.1) Suppose that there are N demonstration trajectories. The N points' variance and mean are calculated empirically:

$$\bar{\mu} = \frac{1}{N}\sum_{k=1}^{N} \bar{x}_{rel,j}(T_k) \quad (4)$$

$$\sigma^2 = \frac{1}{N}\sum_{k=1}^{N} ||\bar{x}_{rel,j}(T_k) - \bar{\mu}||^2 \quad (5)$$

(1.2.2) If the variance is below a threshold, which is scaled according to the entire demonstration workspace size, then the permutation $(T_1, T_2, ..., T_N)$ is taken as a relation interest point which obeys (2).

This RIP is one of many that may be used as a search template. Each type of RIP is really a hypothesis of what the user is trying to demonstrate in some point of the demonstration. He may not know himself what he is showing.

The more RIPs that are provided the more behaviours the system can capture. There is, however, a trade-off. Step 1.2.1 and 1.2.2 of the search algorithm above will execute I^N times where I is the number of points sampled in each trajectory. This scale of algorithmic complexity is very costly and may grow substantially when searching for a complex RIP. Heuristics may be used to narrow the proposed algorithm's search.

2.1.2. Culling of RIPs

Once a set of RIPs is extracted from the demonstration the next step is enforcement of temporal consistency. We assume that each demonstration consists of subtasks performed in the same order. Consider an example list of RIPs in Table 1.

RIP 1 occurs at time 223 in demonstration trajectory 1 and 221 in trajectory 2. RIP 2, however, occurs at time 248 in trajectory 1 and 186 in trajectory 2. We have assumed that each demonstration's component milestones were reached in the same order, thus these two RIPs contradict. To resolve these contradictions the approach taken is to select RIPs according to a metric and eliminate weaker RIPs that contradict.

The metric used took the following form. The number of times a particular RIP occurs in each trajectory is divided by the total corresponding demonstration time. This normalizes the time so that each may be compared to see whether they occurred at more or less the same point in each demonstration. The metric was calculated by determining a mean of all these normalized values and then calculating the empirical variance. The smaller the variance the greater chance they occurred at more or less the same relative time in each demonstration and thus the more likely the RIP.

An example result of the 2 stages of characterization discussed may be seen in Fig. 3. The RIP locations are plotted on one of the recorded trajectories in Fig. 2.

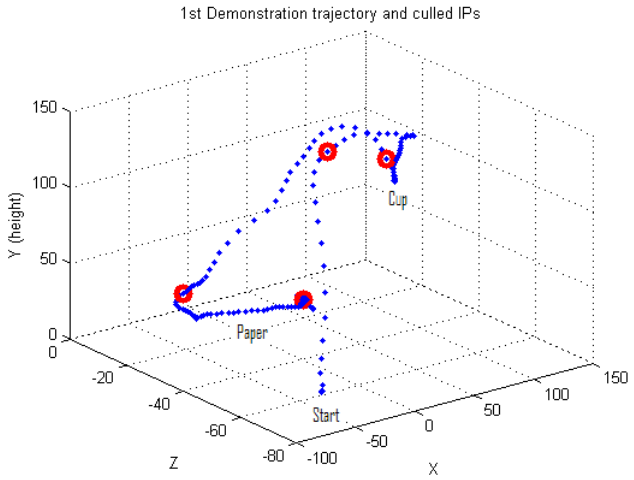

Figure 3: The result of culling RIPs with temporal inconsistency. Red circles indicate RIP locations with the path in blue.

2.1.3. Segmentation and normalization of demonstration trajectories

The previous step provides a set of RIPs that are used to segment each of the demonstration trajectories. The set of segments between a pair of RIPs will have different starting and end points depending on the location of the fiducials in each element's demonstration. They may also have different orientations. It is thus necessary to normalize the set. The scheme used procedes as follows. Every sample subtrajectory is shifted so that the starting point is at the origin. The subtrajectory is then rotated so that the end point is along the x-axis. Finally, the x-coordinate of each sample point is scaled so that the end points sit at 1 on the x-axis.

2.1.4. Gaussian mixture modelling of subtrajectories

In the next step of the architecture, the sets of corresponding normalized subtrajectories (between the same RIPs), such as from the start to the first RIP in each demonstration, are combined and converted into a probability density function. Spatial coordinates as well as time are modelled. The model used is the Gaussian mixture model and the algorithm applied to fit the model is a specialized expectation maximization (EM) algorithm. A Gaussian mixture model (GMM) of a density function takes the following form

$$G(\bar{x}) = \sum_{i=0}^{M} \pi_i N(\bar{x}|\mu_i, \Sigma_i) \quad (6)$$

where N is a Gaussian probability density function of M dimensions. For the model to be a valid density function the scaling factors, π_i, must sum to 1. Symbols μ_i and Σ_i represent the mean vector and covariance matrix of mixture component i.

The standard EM algorithm is not well-suited to the task of fitting the subtrajectories to a GMM for a number of reasons. Firstly, the standard algorithm assumes a fixed number of density components. This value can be determined empirically

[10] or through trial-and-error with some best fit metric. Such approaches, however, offer few guarantees. Secondly, densities may fit data poorly inspite of convergence. This arises from the fact that the EM algorithm guarantees convergence to only a local likelihood maxima. Fortunately, there are a number of powerful EM variants arising from the fields of pattern recognition and neural networks. The algorithms SMEM (Split and Merge Expectation Maximization) [9] and CEM (Competitive Expectation Maximization) [10] inspired the version used. It operates as follows:

(1) *Initialization.* A GMM with 1 mixture component is initialized randomly.

(2) *Standard EM algorithm.* The standard EM algorithm is applied until the likelihood of the data using the current GMM converges:

(2.1) Expectation step:

$$p_{k,j}^{t+1} = \frac{\pi_k^t N(\bar{\epsilon}_j|\mu_k^t, \Sigma_k^t)}{\sum_{i=1}^{M} \pi_i^t N(\bar{\epsilon}_j|\mu_i^t, \Sigma_i^t)} \tag{7}$$

$$E_k^{t+1} = \sum_{j=1}^{K} p_{k,j}^{t+1} \tag{8}$$

(2.2) Maximization step:

$$\pi_k^{t+1} = \frac{E_k^{t+1}}{K} \tag{9}$$

$$\bar{\mu}_k^{t+1} = \frac{\sum_{j=1}^{K} p_{j,k}^{t+1} \bar{\epsilon}_j}{E_k^{t+1}} \tag{10}$$

$$\Sigma_k^{t+1} = \frac{\sum_{j=1}^{K} p_{j,k}^{t+1} (\bar{\epsilon}_j - \bar{\mu}_k^{t+1})(\bar{\epsilon}_j - \bar{\mu}_k^{t+1})^T}{E_k^{t+1}} \tag{11}$$

(3) *Keep covariance matrices positive definite.* The eigenvalues of each component's covariance matrix is kept above some minimum. This ensures that the covariance matrices are invertible.

(4) *Elimination.* Suppose K is the number of parameters required to fully specify a mixture component and S is the number of samples. If a scaling factor, π_i, of any mixture component drops below K/S then that component is destroyed. The idea was inspired by [10].

(5) *Merge.* Any density components with similar covariance matrices and very near means are fused.

(6) *Split.* To determine which density component is best to split the Kullback-Lieber divergence is calculated for each using

$$J(k) = \int f_k(\bar{x}) \log \frac{f_k(\bar{x})}{p_k(\bar{x})} d\bar{x}. \tag{12}$$

In the equation $f_k(\bar{x})$ is a histogram discrete density function and $p_k(\bar{x})$ is the sampled Gaussian density at the histogram bucket centers. Only points within two standard deviations of a density component are used in the calculation. The component with the maximum divergence is split and each new component is randomly initialized in the vicinity of the parent. This step is not executed if it is the last iteration of the algorithm.

The purpose of this step is to encourage densities to cover data that is likely to be generated from a Gaussian density.

(7) *Iteration limit check.* The iteration count is checked to see if the limit is exceeded and, if so, the algorithm is terminated.

(8) *Loop.* If a specified number of algorithm iterations has not been reached go to step 2.

A result from the algorithm is shown in Fig. 5. Each blob represents a mixture component. The ellipsoids illustrate each components standard deviation.

In summary, the characterization process yields a set of RIPs and Gaussian mixture models that characterize subtrajectories between these RIPs.

2.2. Reproduction

The reproduction process proceeds as follows:

2.2.1. Determination of Start and End Points of Subtrajectories

To produce a complete behaviour trajectory the first requirement is for the user to specify the locations of all relevant objects. Using the object locations the most likely positions of all the behaviour model RIPs are calculated.

2.2.2. Generation of Subtrajectories

A likely subtrajectory needs to be generated between every pair of RIPs using the appropriate GMM. To do this, the following algorithm is used:

(1) $T = 0$, where T is the relative time-of-arrival from the starting RIP of the current point.

(2) *Optimize* $p(\bar{x}, T)$, where $\bar{x}$ is the point's location. Use Nelder-Mead to calculate the most likely location of this point given the GMM and that its time is fixed at T. The cost function also optimizes for likelihood of some midpoints between the last and current point. The last point is not changed. This ensures that the optimizer follows a subtrajectory mode and that sequential points do not straddle a region of low likelihood.

(3) *Accept* $\bar{x}$ *if* $p(\bar{x}, T) > L$, where L is a threshold which prevents very unlikely points from adding to the generated subtrajectory. In situations where the GMM components have means away from the current T and small enough temporal variances, $p(\bar{x}, T)$ can be negligible everywhere. These time locations must be skipped.

(4) *Increment* T *if* $T < T_{end}$ *and go to step 2*, where T_{end} is the last points temporal location. Otherwise, terminate and return the set of accepted points.

The optimization problem cannot be tackled with Newton's method and related nonlinear optimizers such as Levenberg-Marqualdt because it does not satisfy their convergence criteria.

A low pass filter is used to smooth the resulting path and then a spline fit provides the ultimate subtrajectory.

3. EXPERIMENTAL RESULTS

The example problem was characterized with the following parameters. The complete SMEM-variant algorithm was run for 45 iterations to characterize each subtrajectory. The maximum number of standard EM iterations was 500.

During reproduction, 100 points where optimized to create a path. Any point with a likelihood less than 10^{-16} was discarded.

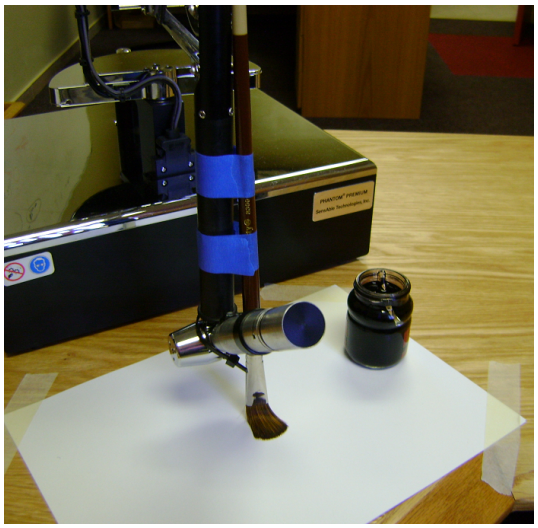

Figure 4: The experimental setup: A cup containing ink, paper and a brush attached to the Sensable Phantom.

3.1. Characterization

Fig. 5 shows three recorded trajectories in the upper plot between two sequential RIPs. A GMM was fitted using the discussed EM algorithm and is also shown in the plot. The yellows blobs are the GMM's components' ellipses of constant standard deviation (1 standard deviation).

The figure illustrates the result of characterizing three subtrajectories with density functions. Although not shown in the plots, each component has a time mean and variance as outputs from the EM algorithm. Components with 'early' means and smaller time variances capture behaviour early in the subtrajectory. This plays an important role in the reproduction phase.

A problem with the scheme is illustrated in the results. Gaussian densities cannot precisely model a density over trajectories. They can only provide an approximation. A precise stopping condition for the proposed EM algorithm is hard to nail down unless one assumes a single modal model and falls back on to the simpler Bayesian Information Criterion or resorts to Bayesians methods in general. This is left for future work.

3.2. Reproduction

In the lower plot of Fig. 5 is the Nelder-Mead generated path alongside the density function. The black dots illustrate intervals of time evenly spaced through the subtrajectory. They are also the points optimized. A cubic spline (red) is fitted through the result of the path planning algorithm to produce the result in the figure. This smooth spline is used to move the manipulator.

An important characteristic of the path is that it follows a particular ridge of high probability and does not skip across areas of low probability. Fig. 6 shows a complete reproduction of a behaviour. Each fiducial (black circle) has been labelled. While demonstrating, the author avoided the lip of the cup and this motion was imitated by this system. It also captured a "dipping motion" in the ink.

The reproduced curves have a 'squared' appearance. This is an artifact of the reproduction method and GMMs. Paths tend to run along the central axis of dominant mixture components. A strategy to correct this could be the use of another type of mixture model with components that can adopt curves.

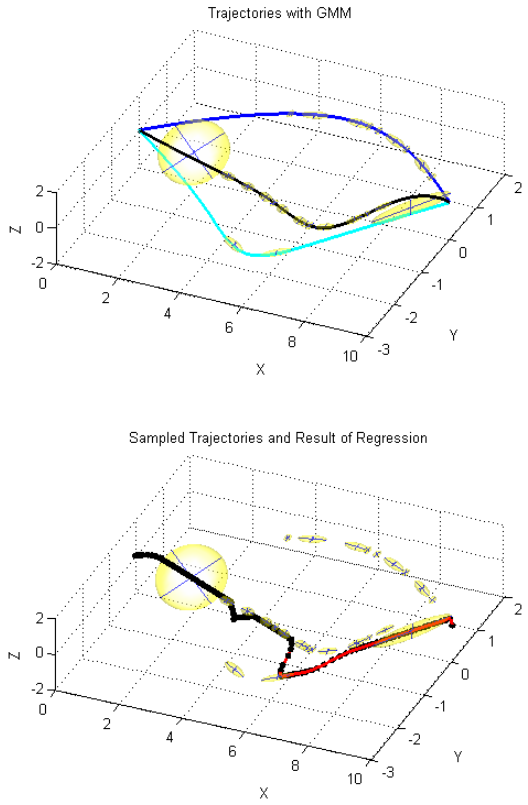

Figure 5: Top) The three recorded subtrajectories in blue, black and cyan with the generated GMM density components highlighted, bottom) A generated path in red with blacks dots denoting intervals of equal time

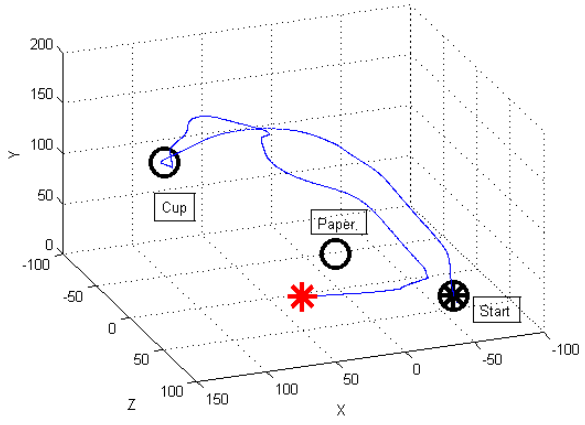

Figure 6: A complete reproduction of the "line drawing" behaviour. The large black circles indicate fiducials and the red star indicates the end of a demonstration.

4. CONCLUSIONS AND FUTURE WORK

An imitation system was proposed that uses relational interest points and Gaussian mixture models to characterize a behaviour. The model extracted can also be used to identify a behaviour. Reproduction is done through nonlinear optimization using a variant of the Nelder-Mead algorithm which produces a highly probable path given the behaviour model. The system was successfully demonstrated on a line drawing task. Although simple, it illustrates that the imitation system will generalize when accuracy in a task is not required and satisfy precise requirements when identified. A great many behaviours would require similar abilities as the example task.

Given that reproduction merely requires the parameters of each GMM and the RIPs with their parameters, it is clear that the imitation system is effectively a behaviour compression system. With the first example the storage space amounts to around 100 floating point double precision variables. This allows a designer to store many taught behaviours in a single system.

For future work the Phantom haptics input device will function as a master to a slave manipulator (such as a Barrett Whole Arm Manipulator). A vision system will be added to capture fiducial locations.

As mentioned earlier, the scheme's major weaknesses are the imprecise stopping condition of the EM algorithm and the fact that RIPs are not continuous. A continuous spatial constraint which captures regions of trajectories would produce smoother reproduction results. These problems are currently being investigated.

5. ACKNOWLEDGEMENT

The author would like to thank Prof. M. R. Kaimal for his advice and help in implementing the SMEM-variant algorithm.

6. References

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

[2] S. Muench, J. Kreuziger, M. Kaiser, R. Dillmann, Robot Programming by Demonstration (RPD) - Using Machine Learning and User Interaction Methods for the Development of Easy and Comfortable Robot Programming Systems, *In Proceedings of the 24th International Symposium on Industrial Robots (ISIR '94)*, 1994.

[3] H. G. Mayer, D. Burschka, A. Knoll, E. U. Braun, R. Bauernschmitt, R. Lange, Human-Machine Skill Transfer Extended by a Scaffolding Framework, *In Proceedings of IEEE Int. Conference on Robotics and Automation*, pp. 2866-2871, 2008.

[4] 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.

[5] S. Calinon, F. Guenter, and A. Billard, On Learning, Representing and Generalizing a Task in a Humanoid Robot, *IEEE Transactions on Systems*, Man and Cybernetics, Part B, Special issue on robot learning by observation, demonstration and imitation, 37:2, pp. 286-298, 2007.

[6] S. Calinon, Continuous Extraction of Task Constraints in a Robot Programming by Demonstration Framework, Thesis, EPFL, Lausanne, Switzerland, 2007.

[7] A. Levas, M. Selfridge, A user-friendly high-level robot teaching system, *In Proceedings of IEEE Int. Conference on Robotics and Automation*, pp. 413-416, 1984.

[8] S. Quinlan, and O. Khatib, Elastic Bands: Connecting Path Planning and Control, *In Proceedings of IEEE Int. Conference on Robotics and Automation*, pp. 802-807, 1993.

[9] N. Ueda, R. Nakano, Z. Ghahramani and G. E. Hinton, SMEM Algorithm for Mixture Models, *Neural Computation*, vol. 12, no. 12, pp 200-, 2000.

[10] B. Zhang, C. Zhang, and X. Yi, Competitive EM Algorithm for Finite Mixture Models, *Pattern Recognition*, vol. 37, pp. 131-144, 2004.