

Towards One Shot Learning by Imitation for Humanoid Robots

Yan Wu, and Yiannis Demiris, *Senior Member, IEEE*

Abstract—Teaching a robot to learn new knowledge is a repetitive and tedious process. In order to accelerate the process, we propose a novel template-based approach for robot arm movement imitation. This algorithm selects a previously observed path demonstrated by a human and generates a path in a novel situation based on pairwise mapping of invariant feature locations present in both the demonstrated and the new scenes using a combination of minimum distortion and minimum energy strategies. This One-Shot Learning algorithm is capable of not only mapping simple point-to-point paths but also adapting to more complex tasks such as those involving forced waypoints. As compared to traditional methodologies, our work require neither extensive training for generalisation nor expensive run-time computation for accuracy. This algorithm has been statistically validated using cross-validation of grasping experiments as well as tested for practical implementation on the iCub humanoid robot for playing the tic-tac-toe game.

Index Terms - movement imitation, path planning, grasping, learning by imitation, tic-tac-toe

I. INTRODUCTION

Since the mid 1980s, programming by demonstration (PbD) has emerged as a promising research topic in robotics due to its relative merits over traditional methodologies [1]. An increased interest in learning algorithms that will equip robots to learn by imitation of actions from both humans and other robotic agents has resulted in many research directions within the area. For instance, a hierarchical model [2] was proposed in the domain of human-robot interaction (HRI) for attention and perception, while some other research focused on skill teaching methodologies ([3], [4]) using a range of techniques. Many of these research directions can be encapsulated by the 5 “W”s of imitation - namely who, where, when, what and how. Very often, it takes numerous trials of learning for a robot to generalise each of these abstract aspects in an action.

In an attempt to reduce the number of trials involved in learning, we propose an algorithm to address the “how-to” question in imitation. We introduce a novel computational model for learning path planning by imitation which makes use of a fundamental idea in plan adaptation - the presence of invariant feature points in both the demonstration and a given situation - to generate a motion path for the new scenario. The following sections of this paper will present some related work to our approach followed by the detailed description of the methodology. Experimental results to validate the

statistical fitness of the algorithm using data from human demonstrations will be presented before discussions on an experiment to implement the algorithm for a real-life tic-tac-toe game on a humanoid robot.

II. RELATED WORK

For a given observed trajectory, in order to reproduce it in an unseen situation, simple copying does not always work well. Furthermore, in a new situation, additional constraints might be present, such as forced waypoints that require plan adaptation along the imitated route. Thus, a robot must have the ability to adapt in addition to generalise. Research work in the area focused on approaches either to produce a new path based on a subset of competent tasks to accommodate additional constraints [5], or to generate a unique yet exact corresponding imitation of the previously demonstrated trajectories ([6], [7]).

Most imitation algorithms require several demonstrations of a single task to work successfully ([5], [8]). This tedious and time-consuming process is not favourable to human demonstrators, especially in emergency situations. To speed-up the learning process and reduce fatigue in giving demonstrations becomes crucial in future robotic advancement. Thus, One-Shot Learning, a popular niche area in machine learning might help to address this issue. However, limited literature[9] has been found to address this issue. Some of such works focus on explanation-based methods[10] which are slightly difficult to be implemented on physical robots.

In [11], the authors argue that paradigms generalising observed behaviours into a set of intrinsic complex model parameters limit the ability of user interaction after demonstration. However, many state-of-the-art paradigms in path imitation are in favour of such models. For instance, in algorithms that use Gaussian Mixture Models[1], the generalised parameters are the weights with associated Gaussian parameters. These algorithms deter direct user interaction/intervention, as the abstract meaning of these parameters is not easily manipulable by users. For example, when a demonstrator accidentally perform a wrong movement, instead of amending the model parameters, the user has to either redo all demonstrations or perform many more correct ones.

To address the above issues and the “how-to” question, a path imitation problem can be perceived as a plan adaptation in which the demonstrated path with a set of environmental features is projected into a warping space. Based on the preservation of the spatial relationship between the imitated and the original features in that space, an imitated path can be generated. Such way of morphing a scene in the current

Yan Wu and Yiannis Demiris are with Department of Electrical & Electronic Engineering, Imperial College London, United Kingdom {yan.wu08, y.demiris}@imperial.ac.uk

context into a new scene exists in the field of both image processing and motion plan adaptation in which energy involved in morphing is minimised ([12], [13]).

III. PROBLEM FORMULATION

In this work, we consider the general case of path planning in a 3-D environment inferred from the demonstration of a similar task. We assume that all required input features are observable from vision, i.e. in our case a pair of stereo cameras. Path planning for a planar task can be thus considered as a special case of this work which requires only one camera.

The ultimate aim of the algorithm is to produce a desirable path for a given scenario. Generation of the path should be an inference from a past demonstrated case. Furthermore, not only should the algorithm be able to generate such path with great level of stability, but more importantly have some resemblance to the path produced by human under similar circumstances.

For a given demonstration viewed from each of the pair of stereo cameras, we describe the motion path as a set of p discretised spatial feature points $\mathbf{m}_l : (x_l, y_l), l \in \{1 \dots p\}$ in the time series images. We also assume that the target destination in the scene can be fully described by a set, F of n point-like features, where each is described by $(\mathbf{a}_i, \mathbf{A}_i), i \in \{1 \dots n\}$. While \mathbf{a}_i represents the Cartesian coordinates of the feature point, \mathbf{A}_i encapsulates additional information that will help to match invariant points of similar objects/targets, such as SIFT[14] features and textual features. We refer to such a given demonstration as a *template* and a new situation as the *task*.

In a new situation described similarly by a set $F' : (\mathbf{a}'_j, \mathbf{A}'_j), j \in \{1 \dots n'\}$, we assume that there exists a confidence function for correspondence, $f_c(\mathbf{A}_i, \mathbf{A}'_j)$, where

$$f_c(\mathbf{A}_i, \mathbf{A}'_j) = \begin{cases} 0 & \text{if } \mathbf{A}_i \text{ \& } \mathbf{A}'_j \text{ are uncorrelated} \\ 1 & \text{if } \mathbf{A}_i \text{ matches } \mathbf{A}'_j \end{cases} \quad (1)$$

Depending on the context, we can employ different feature mapping algorithms to match \mathbf{A} s and \mathbf{A}' s. We, thus, can identify a maximum k pairs of coordinates in the image space of both the *task*, \mathbf{a}'_j , and one of the learned *templates*, \mathbf{a}_i , where $f_c(\mathbf{A}_i, \mathbf{A}'_j) = 1, k \leq n, k \leq n'$. This k pairs of coordinates should also include the pair of starting positions in the *task* and the *template*.

In a general path planning situation, we might be given more features in both the *task* and the *templates*, such as objects at far sight and textual features of the background. However, there are cases which inclusion of such features generates excessive output distortion which is undesirable. Thus, we should not impose the matching constraints for such features in order to preserve the spatial relationship between the cardinal features and generate a route for the *task*, $\mathbf{m}' : (x, y)$.

In a more complex situation, when the agent is required to pass through waypoints or to avoid en-route obstacles, these extra feature points can also be described by an additional set $F'_x(\mathbf{b}_j, \mathbf{B}_j)$. If F'_x exists in a *task*, these additional feature

points should be considered only when the planned path \mathbf{m}' contradicts \mathbf{b} .

IV. METHODOLOGY

In this section, we will describe, in details, our novel approach for path imitation. Briefly, we first generate a distortional mapping of each spatial point present in the *template* path into a set of possible locations based on minimal distortional energy mapping between the k pairs of coordinates extracted from feature sets F and F' as well as the pair of starting positions. Based on the time series information of the *template* path, the *task* path is created from the cloud of possible waypoints using minimum-energy strategy.

A. Feature Distortion Warping

We define the k Cartesian coordinates \mathbf{a}_i from the *template* as the invariant control points(ICP) \mathbf{P} and the corresponding ICPs \mathbf{a}_j in the *task* \mathbf{P}' . If we can define each mapping from \mathbf{P} to \mathbf{P}' as a function f , to minimise the distortion of feature in space is equivalent to minimise the following energy function [15]:

$$E = \sum_{w=1}^k \|\mathbf{P}'_w - \mathbf{P}_w\| + \lambda E_f \quad (2)$$

where

$$E_f = \int \int_{R^2} (f''_{xx} + 2f''_{xy} + f''_{yy}) dx dy \quad (3)$$

The introduction of the *regularisation parameter*, λ , in (2) is to trade-off between the exact matching of points and the smoothness, which is particularly useful in the presence of noise. According to [15], the mapping function f shown can be defined as:

$$f(x, y) = \alpha_0 + \alpha_x x + \alpha_y y + \sum_{i=1}^k \omega_i \phi(\|(x_i, y_i) - (x, y)\|) \quad (4)$$

where

$$\phi(r) = r^2 \log(r) \quad (5)$$

(5) is a 2nd order polyharmonic spline commonly known as a Thin Plate Spline. In order to ensure that E_f exists, the 2nd derivatives of $f(x, y)$ must be square integrable, i.e. the following three conditions have to be met:

$$\sum_{i=1}^k \omega_i = 0 \quad (6)$$

$$\sum_{i=1}^k \omega_i x_i = \sum_{i=1}^k \omega_i y_i = 0 \quad (7)$$

By letting $\Phi_{ij} = \phi(\|(x_i, y_i) - (x_j, y_j)\|)$ and $\mathbf{v}_i = f(x_i, y_i)$, based on (4) - (7), we can form a linear equation as follows:

$$\begin{bmatrix} \Phi & \mathbf{L} \\ \mathbf{L}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} \mathbf{v} \\ \mathbf{0} \end{bmatrix} \quad (8)$$

where $\boldsymbol{\omega}$ is a column vector of ω_i , $\boldsymbol{\alpha} = [\alpha_0 \quad \alpha_x \quad \alpha_y]^T$ and the i^{th} row of \mathbf{L} , $\mathbf{L}_i = [1 \quad x_i \quad y_i]$.

In [16], it has been shown that the square matrix in (8) is non-singular. Thus, we can define the upper left $k \times k$ sub-matrix of the inverse of this square matrix by \mathbf{M}'_k . It can be shown that $E_f \propto \mathbf{v}^T \mathbf{M}'_k \mathbf{v} = \boldsymbol{\omega}^T \mathbf{M} \boldsymbol{\omega}$. Thus, the optimal solution of $\boldsymbol{\omega}$ and \mathbf{a} with minimum bending energy can be solved either by analytical method or approximation methods described in [17] and [18] depending on the importance of accuracy or computational cost.

For each \mathbf{m}_i in the *template*, there exists $q \geq 0$ mapped coordinates as possible candidates for \mathbf{m}'_i in the *task* defined by (4). As these coordinates are often at sub-pixel level, the minimum energy enforcement is relaxed and thresholded to accommodate neighbouring coordinates.

B. Minimum-Energy Route Plan

Given the time series point clouds of \mathbf{m}' , the goal-directed movement is simply represented by stepping through the variable l , and connect the best points from each \mathbf{m}'_l to form the trajectory. We shall make use of the translational energy as the cost function to derive the *task* path. Thus, the optimisation criterion CE is to minimise this cost function which is proportional to the sum of changes in positions.

$$CE = \sum_{i=2}^p (\|\mathbf{m}'_i - \mathbf{m}'_{i-1}\|)^2 \quad (9)$$

Instead of evaluating the full mesh of discrete energies through steps, if we define $SD_{ij} = \|\mathbf{m}'_i - \mathbf{m}'_j\|^2$, U as a possible waypoint in a given time-stage l , $CE_l(U)$ as the minimum energy from the starting point to the waypoint U , we can simplify the computation into a dynamic programming problem[19]:

$$CE_l(U) = \min_{\text{waypoints } v \text{ in } l-1} \{SD_{uv} + CE_{l-1}(V)\} \quad (10)$$

C. Iterative Plan Adjustment

Recall that we have defined F'_x as the features denoting extra features in a more complex situation. After the *task* path is generated, we will then check if any \mathbf{m}'_i contradicts with \mathbf{b}_j in F'_x . In the case of having forced waypoints, suppose we define the contradiction evaluation Ct as

$$Ct(\mathbf{b}_j) = \begin{cases} 1 & \text{if } \mathbf{b}_j \notin \mathbf{m}' \\ 0 & \text{otherwise} \end{cases} \quad (11)$$

The algorithm should iterate the following steps until all $Ct(\mathbf{b}_j) = 0$:

- 1) Compute $Ct(\mathbf{b}_j)$ for all i
- 2) For any $Ct(\mathbf{b}_j) = 1$, locate the point \mathbf{m}'_i on the *task* path that is nearest to \mathbf{b}_j . Find the corresponding point \mathbf{m}_i in the *template*. Put \mathbf{m}_i into set F and \mathbf{b}_j into F'
- 3) Re-perform the distortion mapping algorithm described above to find a new *task* path

V. EXPERIMENTS

Our planning algorithm was implemented and validated on the iCub (Fig. 1a), a humanoid robot developed by the RobotCub Consortium¹. Two different sets of experiments

were conducted to test the statistical fitness (Experiment A) and practical application (Experiment B) of the algorithm respectively. In both cases, we made use of the SIFT matching algorithm as the confidence function f_c defined in (1).

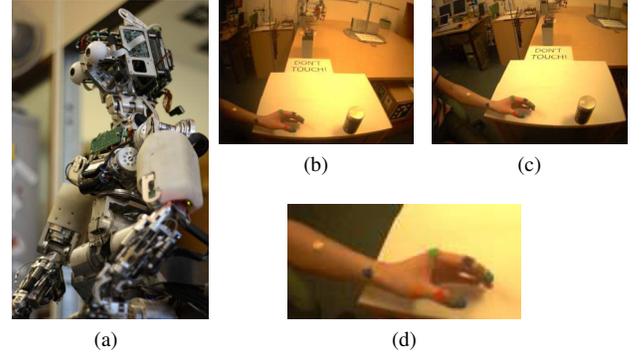


Fig. 1: The experiment set-up for testing the path planning algorithm. The iCub in (a) is developed by the RobotCub Consortium. It has a total of 53 Degrees of Freedom, 32 of which are distributed on the arms. (b) and (c) are an instance of a human subject with markers captured by the left and right cameras of the iCub respectively. (d) shows the locations of markers placed on the left arm of the human subjects in Experiment A.

A. Experimental Setup

The pair of stereo cameras on-board were used to obtain the demonstrated information. In the following reported experiments, the iCub captured the demonstrations at the frame rate of 20Hz and frame resolution of 320×240 pixels (example shown in Fig. 1b & 1c). Markers were placed on human subjects (e.g. Fig. 1d) to track the points of interest.

B. Experiment A

A human demonstrator was instructed to perform a grasp-oriented task while the iCub observed the actions. As extended from our previous work [20], we conducted 5 experimental tasks, which consisted 20 trials each performed by different subjects. This has resulted in 100 experiments for cross-validation of the model against human demonstrations.

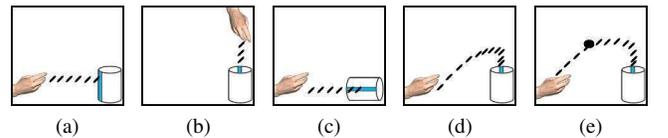


Fig. 2: The sketches of the 5 conducted Experiments. All subjects have been requested to use their inferior arm (in all cases, left arms). The hand positions in the diagrams indicate the starting points of the experiments. The subjects have also been requested to approach the object with their fore-arms orthogonal to the blue strips indicated in the diagrams. The black patch indicated in (e) denotes the waypoint area the subject have to navigate their arms through. The hypothesised paths are denoted by black slashes in the diagrams.

The 5 experiments were designed to benchmark the robustness of the proposed model in various ways, which are described below and illustrated in Fig. 2.

¹www.RobotCub.org

- 1) Experiment 1 shown in Fig. 2a is the most general case of grasping, hypothesised to be most useful for mapping in complex situations.
- 2) Experiment 2 shown in Fig. 2b is intended to test the algorithm with only pure angular rotation of the entire scene.
- 3) Experiment 3 shown in Fig. 2c is designed to test the ability of generalisation of the algorithm into a 3D situation.
- 4) Experiment 4 shown in Fig. 2d is to test the general performance of the warping algorithm.
- 5) Experiment 5 shown in Fig. 2e is to test the robustness of the algorithm in a more complex situation, i.e. in this case with a waypoint.

C. Experiment B

The task of this experiment is to make the iCub to play the tic-tac-toe game by imitating the basic movement of the game. During the training phase shown in Fig. 3a, the iCub was given one single demonstration of how to place a mark in the grid space. In the demonstration, the human subject was instructed to constrain his arm movement on a plane. The iCub was then given a new grid of different size at a completely new location, as shown in Fig. 3b, to play the game with a human until the game is finished. The iCub's arm position was randomly parked some distance above and to the left of the playing board at the end of each move. This parking position would be used as the initial position for the next move. All these measures are to test the robustness of the algorithm to generalise in a 3-D environment. In this experiment, we assumed that the pen was always on the hand of the iCub and the invariant features were the four corners of the cell on the grid and the starting position of the arms.



Fig. 3: **Environmental setup for Experiment B.** (a) shows an instance of the demonstrator drawing a circle in a cell of the A4-sized grid. The iCub was expected to play the game in scene (b), where the grid is 20% smaller, rotated and 20cm above that in (a).

D. Implementation of the Algorithm

As both cameras on the iCub have certain degrees of fish-eye distortion (Fig. 1b & 1c), we undistorted the images with a set of calibration parameters discussed in [21] before processing the captured frames. The marker positions were extracted using an efficient colour segmentation technique proposed in [22]. As the image resolution is limited and sub-pixel data cannot be recorded, uncertainty in extraction process with the present of noise is unavoidable. The least possible amount of uncertainty is thus 0.5 pixel given full

accuracy in extracting the markers. Hence, Gaussian Radial Basis Function (RBF) smoothing with smoothing parameter of 0.5 pixels was applied to the extracted path.

The algorithm with intrinsic parameter $\lambda = 0$ was then set up as we believe that the RBF smoothing should have helped to remove the noise present in the experiments. Thus, any attempt to relax λ might result in the distortion of the optimal path. We also thresholded the distortional energy to accommodate the inclusion of pixel locations up to 2 pixels away from the mapped sub-pixel location.

With the set of predefined intrinsic parameters of the stereo cameras, the two independently generated paths can be integrated to form the final 3-D action path. In Experiment A, the generated paths were then used to cross-validate the results produced by other experimental trials. For Experiment B, during the iCub's term in the game, it generated a corresponding path to place a mark in the intended grid cell. This path was then passed to the inverse kinematics module of the iCub for execution.

E. Statistical Performance Evaluation

Apart from visual inspection, we introduce two performance metrics, namely Mean Squared Difference and Correlation Coefficient, to evaluate the performance of the algorithm quantitatively.

1) *Mean Squared Difference (MSD)*: We make use of MSD to estimate the squared difference between the calculated path and the demonstrated path to gauge how close the generated paths are to human demonstrated ones.

$$MSD = \frac{1}{N} \sum_{i=1}^N \| \mathbf{m}'_i - \mathbf{m}_i \|^2 \quad (12)$$

2) *Correlation Coefficient (R^2)*: Assuming that the proposed algorithm is an estimation of the resulted path generated by human under similar circumstances, R^2 is an indicator of how likely our proposed algorithm can be used to predict paths produced by human.

$$R^2 = \frac{\sum_{i=1}^N (\mathbf{m}_i - \bar{\mathbf{m}}) \cdot (\mathbf{m}'_i - \bar{\mathbf{m}}')}{\sqrt{(\sum_{i=1}^N (\mathbf{m}_i - \bar{\mathbf{m}})^2)(\sum_{i=1}^N (\mathbf{m}'_i - \bar{\mathbf{m}}')^2)}} \quad (13)$$

where $\bar{\mathbf{m}}$ denotes the arithmetic mean of \mathbf{m}_i .

In (12) and (13), both the performance metrics require the input vectors to be of the same lengths. However, in actual practice, we cannot ensure all demonstrations to be completed at the same duration. Thus, we employed the Cubic Spline Interpolation method to lengthen the path with fewer waypoints to match that of the longer one.

VI. RESULTS AND DISCUSSIONS

A. Experiment A

For each trial as an input to the algorithm, we generated 100 paths based on the constraint-mapping into all trials, including the input scenario itself. This has produced a 100×100 matrix of paths for cross-validation.

TABLE I shows the performance metrics for the 100 self-mapping cases, i.e. the input and output constraints are the

same. Assuming a given demonstration is the optimal path, the output from the algorithm should preserve maximally the input path. We can see that in all cases, the confidence indicator is greater than 99% with very low MSD which hints good preservation of paths.

TABLE I: Performances of self-mapping cases grouped in experiments.

Indicators	Exp 1	Expt 2	Expt 3	Expt 4	Expt 5
MSD	25.9	14.4	35.3	31.4	30.6
R^2	0.995	0.998	0.992	0.993	0.993

In TABLES II and III, we grouped the performance indicators according to input/output experiments and take the mean of these indicators. As shown in TABLE III, 88% of the cases have $R^2 \geq 0.7$ which suggests that the path generated by the algorithm are close enough to what human demonstrated. Simple hypothesis testing indicated little statistical evidence in performance difference for $R^2 \geq 0.83$. From TABLE II, we can see that mapping a simpler path, e.g. straight line or without forced waypoints, to any scenario results in closer performance to that of human. Such observation is somewhat expected due to the lack of complete invariant information in the complex case. From these results, we can believe that utilisation of this One-Shot Learning algorithm can reduce the cost and burden of repeated demonstrations, while maintaining high accuracy.

We can also see that the algorithm is capable of mapping paths from Experiments 1-4 accurately to Experiment 5 with the constraint of an additional waypoint. This implies that the IPA algorithm works sensibly in such situations. Thus, we believe that with similar strategy, obstacle avoidance can also be executed with IPA by carefully moving the path point to the neighbour of the obstacle with lowest bending energy. As such, this algorithm can also be treated as an incremental learning paradigm in path planning to accommodate additional constraints which is in-line with human learning pattern [23]. This is probably why mapping

TABLE II: The averaged Mean Squared Difference for mapping from one experiment to another. Columns indicate input while rows indicate output.

	Exp 1	Exp 2	Exp 3	Exp 4	Exp 5
Exp 1	184	312	1158	7165	8244
Exp 2	120	70	1526	3041	5105
Exp 3	781	2785	536	891	2848
Exp 4	1512	645	1216	193	332
Exp 5	1119	865	1391	464	288

TABLE III: The averaged Correlation Coefficients for mapping from one experiment to another. Columns indicate input while rows indicate output.

	Exp 1	Exp 2	Exp 3	Exp 4	Exp 5
Exp 1	0.960	0.949	0.819	0.453	0.402
Exp 2	0.993	0.984	0.932	0.751	0.710
Exp 3	0.864	0.482	0.896	0.793	0.752
Exp 4	0.773	0.827	0.868	0.967	0.941
Exp 5	0.835	0.817	0.874	0.890	0.962

from complex cases (Exp 5) back to the simplest form (Exp 1) does not yield good results due to the lack of precise constraints in the *task* space corresponding to that in the *template* space.

There are inevitable cases that the algorithm does not reproduce a good path most likely due to the quality of the demonstrated path. Additional layer of algorithm can be added to generalise repeating templates to address such problem. It is also interesting to note that Exp 3 yields high R^2 with high MSD, which implies it might not perform better than others. On the other hand, as the templates are stored as a set of path waypoints and invariant features, users will have the flexibility to understand the underlying contexts and make necessary adjustments such as removing templates having high MSDs without retraining the model.

B. Experiment B

Fig. 4a captures the camera view of the demonstrated path, while Fig. 4b shows the generated paths in the iCub’s coordinates system for execution. We can see from Fig. 4a, the human hand started from the far left and drew a circle on the right before leaving the board vertically in space. This has been well-preserved by the algorithm in the attempt to perform the same task elsewhere in space as shown in Fig. 4b. Although there are some minor imperfections in terms of shape preservation, nevertheless the algorithm demonstrated great level of stability with the board being shifted to a new location, and the arm being relaxed from planar movement. And these imperfections can be reduced by introducing more invariant feature points.

At every move, one of these generated paths was then passed to the iCub’s inverse kinematics module for execution. The sequence of drawings performed by the iCub is shown in Fig. 5. All the symbols marked by the iCub during the game were fairly accurately placed in the correct cell. Comparing to the circles generated by the algorithm shown in Fig. 4b, it appears that the drawn circles are much less smooth. This is likely due to the path generated by the iCub’s inverse kinematics module which is not the shortest distance between two points. However, this does not affect the discrimination between the 2 different symbols in the game.

VII. CONCLUSIONS

We have presented a One-Shot Learning algorithm for robot path imitation in this paper. This algorithm has been implemented and statistically validated using cross-validation results from the grasp-oriented paths demonstrated by human subjects. It has also been implemented to equip a humanoid robot with the capability to play the tic-tac-toe game without being constrained to the same location. The experimental results show that this generic algorithm is capable of reproducing satisfactory path by imitating simple tasks. However, the experiments have been conducted with assumptions, such as sufficient invariant feature points were given for mapping and the destination features were static. Computationally, this algorithm with the implementation of the approximation method is an $O(n^3)$ problem. This

