Novelty Based Learning of Primitive Manipulation Strategies

Benjamin Border, R. Andrew Russell
Intelligent Robotics Research Centre,
Monash University, Melbourne, Victoria, Australia
benjamin.border@monash.edu, andy.russell@monash.edu

Abstract

A novelty based learning system which self generates primitive actions that can be used to form more complex manipulations is presented. In the field of intelligent robotics research, manipulation capability is important as it enables complex, yet delicate and precise interactions with the environment. Thus far, manipulation research has made only minor progress in developing general manipulation techniques. In order to provide the flexibility to adapt to unstructured environments, robotic manipulation systems will require an autonomous learning capability. Conventional machine learning algorithms, such as various forms of neural networks, reinforcement learning and decision tree's, have provided good results in previous research for relatively simple tasks. However for increasingly more diverse and complex tasks, these algorithms lack adaptability and expandability and as such often yield poor performance. A relatively new field of machine learning for intelligent robotics modelled on human behaviour is evolving which has promising potential for manipulation applications. Motivated or novelty based learning uses interesting occurrences or outcomes to direct and focus learning towards such observations of interest. In this project this concept is applied to a simplified robotic manipulation system so that primitive actions can be learned. It is intended that these primitive actions will then be combined to form more complex actions to complete a given task.

1 Introduction

Manipulation of objects is a fundamental and deceptively easy task for humans. For robots, this task is far from trivial due to the non-linearity of manipulator joint kinematics, the high performance requirement of tactile sensors and the necessity to generalise actions within the working environment. Learning to manipulate is a process which takes many years for humans, yet in robotics it is often desired for learning to be completed in a much shorter time frame. This requires some form of redundancy reduction, simplification and generalisation to reduce the problem down to a more basic level.

Over the past several decades robotic manipulation has been researched in conjunction with tactile sensing technologies. This focus is primarily due to tactile sensor technology being the main application. In general, research was focused on very specific aspects of manipulation such as touch mediated rolling, slip detection, surface exploration and grasp control. The lack of significant progress in general manipulation research is often ascribed to insufficient tactile sensor performance and sensor size restrictions [Jacobsen, *et al.*, 1987]. Consequently there has been very little research into intelligent robotic systems which can learn to manipulate objects at a fundamental level, with the ability to generalise and adapt within their environment.

The conventional approaches to intelligent robotic learning include reinforcement learning, neural networks or decision trees. These approaches provide good results for simple or singular tasks. However, when more diverse and complex tasks are attempted the conventional approaches are not capable of providing satisfactory performance. An example in the case of reinforcement learning is when the task becomes more complex, the number of trials required for learning increases dramatically. This reaches a point where physical robots cannot be used and only simulations can perform the required hundreds of thousands of trials [Kaelbling, 1993]. Furthermore, increasingly complex tasks, such as manipulation, require the learning system to be adaptable and have the potential to expand its abilities. Enabling the system to generalise the information it gathers to perform diverse tasks with differing conditions and goals.

When analysing a manipulation task performed by a human, it can be observed that it can often be broken down into a number of fundamental actions. Many of these fundamental actions are common to a range of complex manipulations and form a generic library or table of simple actions. These actions can be combined and used to perform more complex tasks. Speeter called these fundamental actions primitives and implemented them by storing the associated positional movements in a matrix, which describe the motion [Speeter, 1991]. The work conducted by Sungmoon et al. [Sungmoon, *et al.*, 2010]

incorporated a form of primitive generation. However, the manipulator system was very basic and limited to a small area. They used a concept called visual attention shifts to identify 'interesting' or 'novel' visual features such as colour, orientation, scale and symmetry. Such features can be used to focus learning on an object and generate goals for the system. This method is good for identifying possible interesting objects to manipulate, however an assumption is made that if the object does not appear to be interesting, then it probably isn't. Objects which may provide the most interesting result from manipulation may in fact not appear visually interesting.

Such an approach is proactive and requires either a higher level intelligence to discern potentially interesting stimuli, or the system must be given prior knowledge. Therefore, instead of using learning to focus solely on stimuli which appear to be interesting, a different reactive approach focuses on interesting results of actions performed by the system. The system can learn what is interesting within its environment by focusing on observed interesting phenomenon resulting from interaction. There are a number of motivated learning methodologies in the literature. More recent examples include work by Merrick and Maher who proposed a method where a system was motivated to learn from the occurrence of interesting events [Merrick and Maher, 2009], Xinyu and Minghai use novelty generated from a visual environment to learn scene recognition [Xinyu and Minghai, 2011] and Raif and Starzyk discuss a motivated reinforcement learning system, its improved performance and reduced learning time compared to standard reinforcement learning [Raif and Starzyk, 2011]. All of these use a reactive approach for motivated learning and demonstrate the effectiveness of the learning technique.

In many learning scenarios supervised learning is used, where the user knows the solution and shows the system how to complete the task. The system then will learn to imitate what it has been shown. Whilst this is acceptable for some learning problems, when dealing with unstructured environments it is preferable to have the robot learn a task independently. Having to show the robot system how to complete a task restricts the potential for the robot to adapt to unknown and changing environments. Therefore, in this paper an unsupervised novelty based approach is considered for learning manipulations.

2 Manipulation Learning

In this section, an alternative method for creating primitives based on the concept Speeter presents is discussed in detail in Section 2.1. The methodology adopted to visually identify and classify objects being manipulated is presented in Section 2.2. The motivated learning process is presented in detail in Section 2.3. Finally primitive generation using motivated learning is discussed in 2.4.

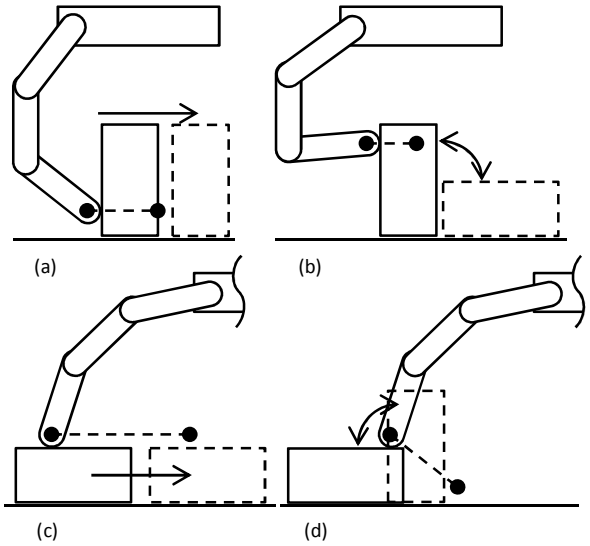


Figure 1 - Example Primitive Actions (a) Sliding; (b) Tipping; (c) Draging and (d) Flipping.

2.1 Primitives

As discussed previously, manipulations in general are complex tasks which often require complex solutions. Attempting to teach a robotic system to perform even simple manipulations via conventional unsupervised machine learning techniques yields poor results. This is due to the number of possibilities of input and output values and the time required exploring all valid possibilities. Neural networks are a good example of such a scenario. As a larger variety of manipulations are desired to be learned, the required number of neuron units to successfully learn a large range of nonlinear actions is also increased, hence requiring an increased number of training cycles [Russell, 2010]. Furthermore without some form of generalisation the learned action would only be useful when all initial conditions were similar to the required action. Consequently the object must be placed in the same location and the goal must be the same.

To simplify joint motion, a method was proposed by Speeter [Speeter, 1991] who developed a process he called motor primitives as a means to control simple actions of a robotic hand. These primitives were comprised of a matrix of joint angle values describing the motion of the primitive whose values were manually input into the matrix. An example of primitive generation to perform manipulation tasks, while using machine learning, is given in the work of Fuentes and Nelson [Fuentes and Nelson, 1997]. A modified version of the evolution strategy was used, generating the primitives which were successfully implemented. This process, while capable of generalised use over the workspace, is limited as the system does not know its own kinematics. Hence it does not know how to move itself prior to learning primitives and must go through the process of learning primitives again, despite even small changes in the kinematics.

Instead of using joint angles for primitive generation, this paper proposes an alternative method which uses visual information from a camera system to obtain an end-point position of a manipulator finger. The kinematics is mapped by using a Kohonen Self-Organising Map [Kohonen, 1990]. This is similar, although somewhat simplified approach to the work of Bassi and Kelly and Bruessler et al. [Bassi and Kelly, 1994;Buessler, *et al.*, 1999]. The process is essentially equivalent to storing observed end-points, with corresponding joint angles generated from random movements of the manipulators fingers. When mapping is sufficiently completed the manipulators fingers can be positioned by specifying end-point positions instead of a series of joint angles. This is particularly useful where more delicate manipulations are required. Often the finger end point is the only part of the manipulator in contact with the object being manipulated. It is unnecessary to explicitly control all joint angle values given there are multiple solutions to move to the same end-point position. Some typical primitive actions using point-to-point movement are shown in Figure 1. To create a primitive, an action is conducted on an object by the system and the change in end-points is stored as a vector, instead of a series of joint angles in a matrix.

2.2 Object Classification and Generalisation

Visual feedback, while not essential for many manipulations, can provide a fast method to determine the type of object the system is attempting to manipulate and the results of the manipulation [Kroemer, *et al.*, 2011]. Therefore, a simple object classification system has been implemented using the OpenCV library to locate and identify objects in the experimental workspace. A threshold is applied to the image from a webcam, which is then segmented and contours are found. A polygon approximation algorithm is then used to obtain the object shape. Figure 2 shows the output of the camera system when simple shapes are visualised. It should be noted that curved faces are approximated by multiple straight-line segments. The system stores the polygon shape in an array each time a unique shape is encountered. The shape similarity or uniqueness, is determined by comparing the Hu invariants using the 'matchShapes' function in OpenCV, which is shown in Equation 1, such that the sum of the ratio of difference is obtained.

$$I(A,B) = \sum_{i=1..7} \frac{\left|m_i^A - m_i^B\right|}{\left|m_i^A\right|} \qquad (1)$$

Where A is the current shape and B is a stored shape

$m_i^A = sign(h_i^A) \cdot log h_i^A$
$m_i^B = sign(h_i^B) \cdot log h_i^B$

h_i is the Hu invariants of A and B

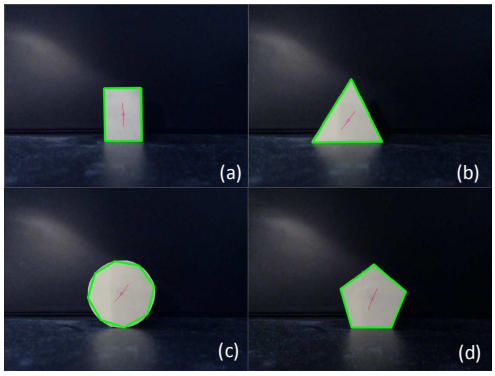


Figure 2 - Example Shapes (a) rectangle, (b) triangle, (c) circle approximated to an octagon and (d) pentagon.

A threshold is then applied to the output *I* from the 'matchShapes' function. This threshold was determined by trial and error until an acceptable value was found.

This method of classification enables the system to generalise all manipulation information about each shape it has encountered. Therefore, the system is not told what each shape is by a user. It only knows they are different, allowing objects that are the same basic shape, yet of different size and orientation within the workspace, to be associated and grouped with any learned primitives. For simplicity and the ability to generalise, an assumption is made that all actions will be valid for every particular shape the system matches an object to. This is not always true as a shape's size and mass can change the effect a primitive action has on the object. However, for the range of simple shapes supplied to the system while testing, this will serve as a first approximation.

2.3 Motivated Learning

Conventional machine learning of all types, use a criterion to determine when learning should be stopped. This is typically when a specified number of iterations are completed or the learning error falls below a threshold. A greedy or highest reward method is often used to determine the course of action for each decision when the system is undergoing learning. This can lead to the system missing potentially interesting or useful information to learn.

Conventional reinforcement learning uses goals to determine if the actions taken were good, which often requires user input to specify a goal, or a complex higher level learning system to generate goals. Motivated learning is similar in basic concept in that good behaviour is rewarded. However, motivated learning uses the results of the actions taken by the system to determine if those actions were 'interesting' and as a consequence good, instead of using pre-defined or generated goals to evaluate how good an action was. These results are evaluated and classified as novel or interesting by various means, such as identifying patterns or anomalies as in the work of Merrick and Maher [Merrick and Maher, 2009], object colour,

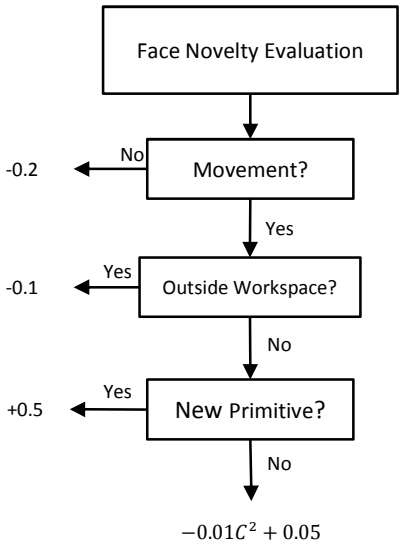


Figure 3 - Novelty Evaluation. Demonstrates how the face novelty is modified when an action is evaluated.

orientation, scale and symmetry as Sungmoon, et al. implemented [Sungmoon, *et al.*, 2010] or internal motivation from scene learning implemented by Xinyu and Minghai [Xinyu and Minghai, 2011]. Furthermore, as the learning focus is not restricted to the current 'best' solution found by a greedy strategy all learning possibilities can be explored. This is particularly useful where certain conditions such as a tipping point of on object are required to be met before a positive outcome occurs.

To implement this concept of motivated learning for manipulation it is necessary to first define some terms.

Interesting - In this learning system, the concept of 'interesting' is related to changes in the position and orientation of the object that is being manipulated. Thus an action which results in a change in pose which has not been observed before, is considered to be very interesting. Conversely, an action which has been observed repeatedly is considered uninteresting.

Novelty – The 'newness' of the current action which the system has observed. Novelty values range from [0,1] where 0 is considered old and of low priority to the system and 1 is considered novel and of high priority to the system. An interesting action will increase this value, whereas an uninteresting action will decrease this value.

Object Novelty – How interesting the current working object is. This is comprised of the novelty of each face of the object as shown in Equation 2.

$$N = \frac{\sum_{j=1..m} n_j}{m} \quad (2)$$

Where m is the number of faces of the object
n_j *is the novelty of face j of the object*
N_i *is the novelty of the object*

Object novelty is used to determine how long to perform trials on the object the system currently has focus on. When this value falls below a threshold, the system becomes 'bored' with the current object.

Face Novelty – The level of novelty an object face has on the system. The face novelty is used to determine which face to work on when performing primitive trials. Initially, each face starts with a novelty of 0.5 – neither interesting nor boring.

When the system evaluates the result of an action, there are several outcomes which result in the adjustment of the face novelty variable. Figure 3 illustrates the process. A new primitive generates a highly positive result as a new motion has been discovered by the system. A value of 0.5 is added to the current face novelty to reflect the new discovery. If a similar movement of the object is observed where no new primitive is created and there is already a similar primitive, the system adjusts the face novelty as shown in Equation 3. Changing the values a and b will affect the speed with which the system becomes bored with the current face.

$$n_t = n_{t-1} - aC^2 + b \quad (3)$$

Where C is the number of times a similar movement has occurred and
a = 0.01,
b = 0.05

If the finger reaches the target position, yet no movement is detected from the object, an unexpected outcome has occurred which does not provide an interesting result. Hence, a value of 0.2 is subtracted from the current working face novelty. Finally, if the finger is unable to move to the required target position, the face novelty value is decremented by 0.1. An exception being, that if the object has moved and is consequently outside the bounds of the workspace of the finger, the previously discussed evaluation outcomes are tested for and the face novelty is adjust accordingly.

2.4 Generating Primitives from Interesting Occurrences

A general primitive generation function has been developed, which has built-in actions analogous to human exploration. The process flow is shown in Figure 4. The function has been designed such that the finger has an 'instinctual' tendency to push the surface of an object face at various points. The face novelty discussed in section 2.3, is used to determine the most interesting face and the system subsequently perform actions on it. The finger will continue to attempt to push the surface point until either of

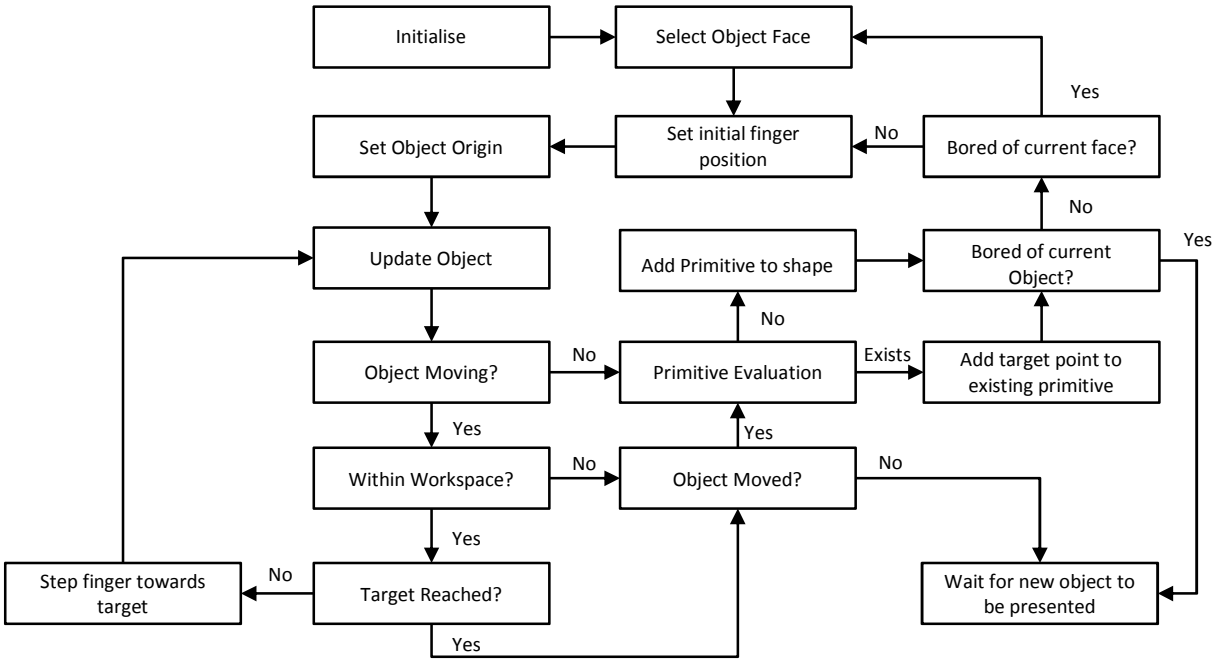


Figure 4 – The process flow of the primitive generation function.

the following conditions are met: the object stops moving, the target point is unreachable by the system, or the finger reaches the target point but no movement was observed.

The primitive data structure consists of a normalised object displacement vector, containing the Cartesian coordinates of the object in the image plane, as well as the objects orientation; the face target position as a ratio, where a value of 0.5 is the mid-point of the face and a value of 0 or 1 are the end points; and lastly, the original object position and orientation before the action was performed. The shape data structure contains the vertices of the shape, the face novelty, the object novelty and the primitive list. Hence, the relationship can be summarised as follows in Equation 4 to 11, where S is the shape dataset.

V is the shape descriptor and contains m vertices

$$V = \{v_1, v_2 \dots v_m\}, \qquad V \in S \tag{4}$$

where

$$v_m = \{x_m, y_m\} \tag{5}$$

As described in section 2.3, N is the object novelty and F is the face novelty.

$$F = \{n_1, n_2 \dots n_j\}, \qquad F \in S$$

and (6)

$$N \in S$$

The primitive dataset P is structured as follows

$$P = \{p_1, p_2 \dots p_k\}, \qquad P \in S \tag{7}$$

$$p_k = \{R_k, T_{k,} O_k, f_k\} \tag{8}$$

Where T_k *is the face target point*
f_k *is the object face index*

R_k *is the normalised object displacement vector*

$$R_k = \{x_k^n, y_k^n, a_k^n\} \tag{9}$$

O_k *is the original object vector*

$$O_k = \{x_k, y_k, a_k\} \tag{10}$$

The shape dataset S is as follows

$$S = \{s_1, s_2 \dots s_i\} \tag{11}$$

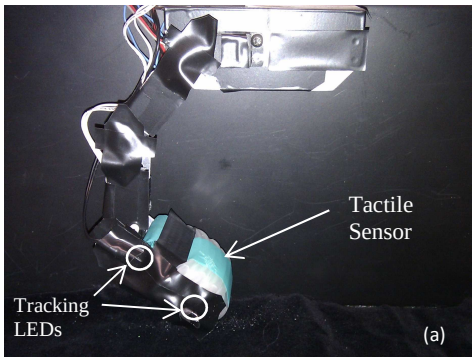
Where i *is the number of shapes*

When an action is completed and evaluation is conducted, the normalised object displacement of all currently recorded primitives for the detected shape, are compared to the current normalised object displacement using the dot product as shown in Equation 12.

$$m = R_k \cdot Q \tag{12}$$

where m *is the directional match*
Q *is the current normalised object displacement vector*

If the vectors have a similar direction, such that $m > 0.85$ and the face indexes match, then the target point is added to the similar primitive target point list, otherwise a new primitive is created. The action of the finger is not recorded



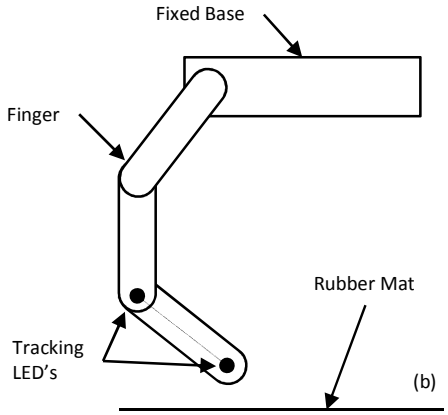


Figure 5 - System configuration and workspace view. (a) shows the finger in the environment and (b) illustrates a frontal view.

as it is a linear motion of variable length. This allows the system to determine how far to push, as only a partial primitive action may be necessary to complete a task. Furthermore, the target points can be used to determine the parts of the object that can be pushed which result in the primitive successfully completing.

This algorithm could potentially be applied to many systems which are required to manipulate their environment. For example, a mobile robot which has an overhead camera monitoring both itself and environmental object movement, can generate primitives and learn how to move the objects within its environment from pushing them around, eventually performing tasks supplied by a user. Furthermore, similar to the implementation presented in this paper, self-organising maps could be used to configure a robotic swarm to cooperatively move and interact with objects. Therefore, a swarm can generate primitives and learn to perform complex maneuvers to complete a task.

A potential issue may arise with this algorithm, where a significant change in the world physics causes actions previously taken to fail as they are unable to be replicated. This is due to the generated primitives no longer having valid orientation information. Hence a new set would need to be generated. For example if the orientation of the workspace changed due to an inclined environment, an action which previously moved an object along the x axes now moves an object along both x and y axis. Whilst

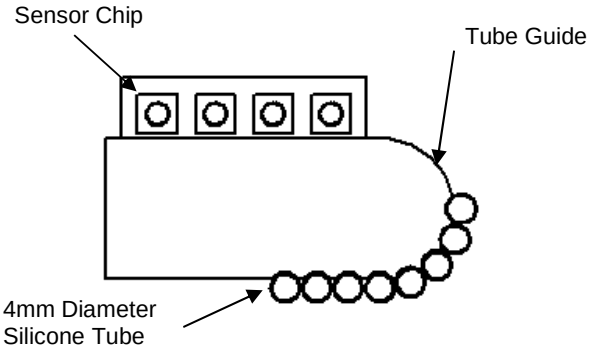


Figure 6 - Sliced view of the tactile sensor. 4 sensors are shown on this side and the remaining 4 are hidden from view behind those.

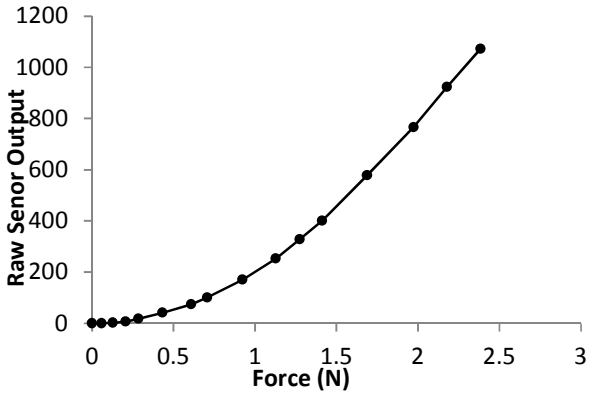


Figure 7 - Tactile sensor single sensor pad sensitivity

the system may be able to compensate, the performance and manipulation time may be affected.

3 Experimental Setup

The experimental setup was designed to be as simple as possible so as to minimise challenges that are not relevant to the key focus of this research. Consequently, the workspace of the system, shown in Figure 5, comprises an essentially 2 dimensional workspace, with a 3 degree of freedom robotic manipulator finger. When fully extended the finger can just touch the base of the workspace, which consists of a rubber mat. The finger is controlled by 3 radio control servos located inside the finger aluminium frames. The finger has a highly sensitive 1 dimensional tactile sensor array embedded into the frame of the fingertip. This consists of 8 sensing pads, constructed from lengths of silicone rubber tube, sealed at one end and connected to an air-pressure sensor chip at the other end (Figure 6). The tactile sensor is currently used to detect contact with objects and to stop the system from pressing too hard on the surface of an object, potentially damaging the fingers servos. It is intended to incorporate the tactile sensor further into the system such that sheer force can be applied to the surface of a face (Figure 1 c.) and tested for potential primitives if a normal force motion cannot be achieved.

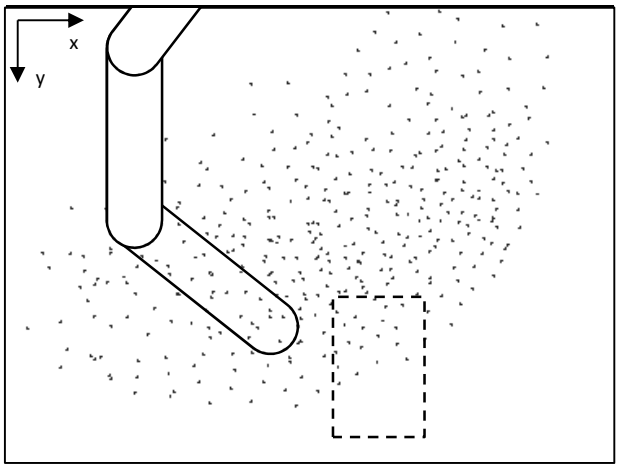


Figure 8 - Finger reachable workspace in camera coordinates. The system is capable of moving the finger end point to any of the points shown. Overlapping points contain different finger orientations. Approximate object size is shown.

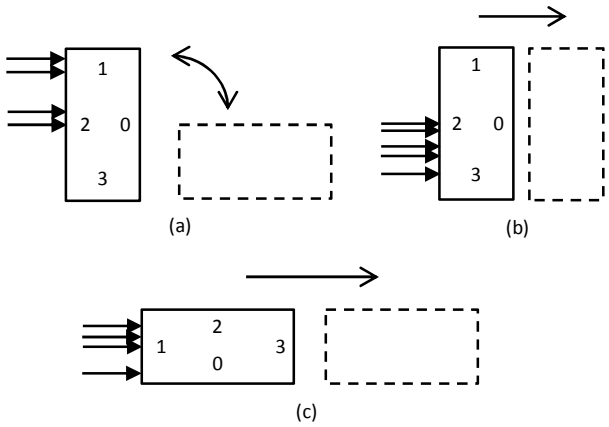


Figure 9 – Generated primitive data where (a) illustrates the object being pushed over and (b) and (c) illustrate the object being pushed along the surface in different orientations.

LED's are placed at the distal finger joint axis and at the fingertip. This allows the camera to track the position and orientation of the fingertip. Black tape has been placed on the finger aluminium frame to reduce specular reflection. The working area of the finger is monitored by a webcam placed approximately 20cm from the finger and elevated approximately 7cm from the mat. The system workspace background was painted black, enabling the white and yellow coloured objects to be easily segmented from the background.

4 Results

Initially, trials have been performed to demonstrate the system's capability to create manipulation primitives. The reachable space is illustrated in Figure 8, where the system is capable of moving the finger end point to any of the points shown. In the first trial, a rectangular object was placed into the workspace of the system. The algorithm, as discussed in section 2.4, was run and the system attempted to apply finger pressure at various points on the object. The results, in terms of object movement and novelty was determined, where novelty was classified as described by Figure 3. After repeated exploration resulting in no more new primitives being discovered, the system became 'bored' and stopped learning. Table 1 contains the shape vertices in camera coordinates, which describes the object detected by the system and the final face novelty after learning ended. Table 2 contains the primitives that were generated until the system became 'bored'. The values *x* and *y* are normal vectors in the direction of movement of the block and *a* is the orientation (-180, 180) scaled to the range of -1 to 1. Figure 9 illustrates the data generated for each primitive. Figure 9(a) corresponds to primitive 0, where the block was pushed towards the top of the face, consequently pushing it over. Figure 9(b) corresponds to primitive 1, where a translation along the x axis occurred until the edge of the workspace was reached. In the case of primitive 2, a partial rotation and translation such that the object was pushed along the ground for a small distance with friction eventually causing the object to tilt. The system stopped halfway through pushing the block over as it had reached the limit of its workspace. This was not able to be repeated during the learning phase, hence only one point of contact as recorded. Figure 9(c) corresponds to primitive 3, where the object is translated similarly to primitive 1, however the action was conducted on face 1 and the initial orientation was different, hence a new primitive generated. It should be noted in the cases where the block was pushed outside the reachable workspace the finger position was reset, the block was manually placed back into the workspace in a random position and orientation and learning was continued.

X	Y	Face Novelty
304	463	0.5
300	340	0
218	341	0
226	462	0.5

Table 1 - Object vertices using camera coordinates. The final face novelty after trials completed.

Primitive	Face Index	x	y	a
0	2	0.769	-0.342	-0.541
1	2	0.999	-0.034	-0.017
2	2	0.644	0.080	-0.761
3	1	0.999	-0.028	-0.019

Table 2 - Primitives created during trials where x, y and a form the normalised displacement vector of the object

Primitives	0	1	2	3
Target Points	0.524	0.752	0.327	0.341
	0.502	0.591		0.591
	0.194	0.624		0.211
	0.083	0.553		0.904
		0.832		

Table 3 - Target points discovered for each primitive.

In total the system conducted 14 trials before becoming 'bored'. The precise target points for each primitive are shown in Table 2. Only face's 1 and 2 were explored due to the object symmetry. Furthermore, faces which were in unreachable positions for the finger during face selection, were temporarily ignored such that erroneous manipulations due to finger path occlusions were avoided. If a second opposing finger was introduced into the system this would not be necessary as the best finger could be selected to perform the operation.

5 Discussion and Future Work

For the next step of this project, it is proposed that once primitives have been generated, a reasoning system will be used which selects the appropriate sequence of primitives to perform a manipulation task shown to it by a user, or which the system generates automatically. Reinforcement learning, decision trees or means-ends analysis will be considered for use in this case, as they are appropriately suited to such tasks. In the situation where the system has classified a shape, yet it is of different size, the Hu invariants as discussed in section 2.2 can be used to transform values to compensate. As a means to identify and segment poorly generated primitives, the success of each trial attempt while learning to sequence primitives will affect later decisions on primitive selection for completing a task. Hence, primitives with high success rates will be considered for use much more favourably by the learning system, than primitives with poor success rates. Furthermore, individual target point information can be used to generalize and create hypothesis of areas which are valid for completing the primitive action. This area can be further refined and tested when the system is learning to sequence manipulations.

The work presented in this paper is a first step towards producing a system that can learn manipulation primitives and assemble them into sequences to perform desired manipulation operations. There are many aspects of the project which require further investigation and development. Such work includes incorporating the tactile sensor into the motivation system, such that force and contact information can be used to increase the quality of primitives. Path trajectory can be implemented to form primitives instead of the current linear point to point method, enabling more complex manipulations. A second finger can be added for two-fingered manipulation and primitive generation. Advanced image processing can be used to track all phalanges of the finger to facilitate more complex kinematic mapping, remove the necessity of the tracking LED's while providing more robust tracking. Eventually, a wrist and arm can be implemented for greater workspace accessibility and capability and finally the system could be brought into the 3-dimensional domain.

6 Conclusion

Building a robotic manipulation system with the dexterity of the human hand has been a goal in robotics ever since it became possible to combine the digital computer with robotic mechanisms. The failure to produce such a system is evidenced by the lack of any industrial process which incorporates a gripper modelled on the human hand and incorporating skin-like tactile sensing. The authors feel that one step towards producing a human-like robotic manipulation system, is to develop systems which can learn manipulation primitives and combine them to execute new manipulation operations. In this project a system which can self-generate primitives using novelty to focus learning has been presented. Preliminary results demonstrate the capability for generating primitive actions. Improvements and future research direction have been outlined in the discussion.

References

[Jacobsen, et al., 1987] S. Jacobsen, I. McCammon, K. Biggers and R. Phillips, Tactile sensing system design issues in machine manipulation, Proceedings of the conference on Robotics and Automation, pages 2087-2096, 1987

[Kaelbling, 1993] Leslie Kaelbling, Learning in embedded systems, 1993

[Speeter, 1991] T.H. Speeter, Primitive based control of the Utah/MIT dextrous hand, Proceedings of the conference on Robotics and Automation, pages 866-877, 1991

[Sungmoon, et al., 2010] Jeong Sungmoon, Lee Minho, H. Arie and J. Tani, Developmental learning of integrating visual attention shifts and bimanual object grasping and manipulation tasks, *Proceeding of the 9th international conference on developemental learning*, pages 165-170 2010

[Merrick and Maher, 2009] Kathryn Merrick and Mary Lou Maher, Motivated Learning from Interesting Events: Adaptive, Multitask Learning Agents for Complex Environments, *Adaptive Behavior*, 17(2):7-17, February 2009

[Xinyu and Minghai, 2011] Qu Xinyu and Yao Minghai, Visual novelty based internally motivated Q-learning for mobile robot scene learning and recognition, *Proceedings of the 4th international congress on Image and Signal Processing*, pages 1461-1466 October 2011

[Raif and Starzyk, 2011] P. Raif and J. A. Starzyk, Motivated learning in autonomous systems, Proceedings on the international joint conference on Neural Networks, pages 603-610, July-August 2011

[Russell, 2010] Russell, *Artificial intelligence : a modern approach*, 2010

[Fuentes and Nelson, 1997] O. Fuentes and R. C. Nelson, Learning dextrous manipulation skills using the evolution strategy, *Proceedings of the conference on Robotics and Automation*, pages 501-506, April 1997

[Kohonen, 1990] T. Kohonen, The self-organizing map, *Proceedings on the IEEE*, 78(9):1464-1480, 1990

[Bassi and Kelly, 1994] D. Bassi and R. Kelly, Neural networks and direct vision to position control of fixed eye robotic systems, proceedings of the *IEEE international symposium on industrial electronics*, pages 82-87, May 1994

[Buessler, et al., 1999] J. L. Buessler, R. Kara, P. Wira, H. Kihl and J. P. Urban, Multiple self-organizing maps to facilitate the learning of visuo-motor correlations, Proceedings of the *IEEE international conference on systems, man and cybernetics*, pages 470-475, 1999

[Kroemer, et al., 2011] O. Kroemer, C. H. Lampert and J. Peters, Learning Dynamic Tactile Sensing With Robust Vision-Based Training, *IEEE transactions on robotics*, 2011