# SIMULTANEOUS HUMAN-ROBOT LEARNING FOR EFFICIENT SKILL SYNTHESIS

A Thesis

by

Mohammad Ali Zamani

Submitted to the
Graduate School of Sciences and Engineering
In Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in the
Department of Computer Science

Özyeğin University
August 2015

# SIMULTANEOUS HUMAN-ROBOT LEARNING FOR EFFICIENT SKILL SYNTHESIS

Approved by:

---

Professor Erhan Öztop, Advisor
Department of Computer Science
*Özyeğin University*

---

Professor Hasan Sözer
Department of Computer Science
*Özyeğin University*

---

Professor Nafiz Arıca
Department of Computer Engineering,
*Bahçeşehir University*

Date Approved: 18 August 2015

*To my wife, for her love and support.*

# ABSTRACT

It is generally expected that robots and autonomous agents will become a part of our daily lives in the coming decades. However, it is not feasible to program robots in advance for all possible tasks using classical robot programming. Therefore, intuitive and easy robot programming is one of the active research areas in robotics. We propose and implement a human-in-the loop robot skill synthesis that involves simultaneous adaptation of the human and the robot. In this framework, the human demonstrator learns to control the robot in real-time to make it perform a given task. At the same time, the robot learns from the human guided control creating a non-trivial coupled dynamical system. The research question we address is how this system can be tuned to facilitate faster skill transfer or improve the performance level of the transferred skill. At the beginning of the skill transfer session, the human demonstrator controls the robot exclusively as in teleoperation. As the task performance improves the robot takes increasingly more share in control, eventually reaching to full autonomy. The proposed framework is implemented and shown to work on some tasks such as physical cart-pole setup, cart-pole balance simulation, and mountain car. To assess whether simultaneous learning has advantage over the standard sequential learning (where the robot learns from the human observation but does not interfere with the control) experiments with two groups of subjects were performed. Moreover, reinforcement learning is applied to model a human demonstrator to verify simultaneous framework. The results indicate that the final autonomous controller obtained via simultaneous learning has a higher performance in the mentioned tasks.

# ÖZETÇE

Yakın gelecekte robotların ve otonom birimlerin günlük yaşamımızın bir parçası haline geleceği beklenmektedir. Fakat, robotları bütün olası görevler için klasik yöntemlerle programlamak mümkün değildir. Bu sebeple, robot programlamayı kolay ve erişilebilir hale getirmek robotik alanındaki aktif çalışma alanlarından biridir. Bu konuda katkı sağlamak amacıyla insan ve robotun eşzamanlı adaptasyonunu temel alan insan etkileşimli `döngüde insan' robot beceri sentezi çerçevesini sunuyoruz. Bu çerçevede, aslında öğretici olan kişi robotu control etmeyi ve verilen bir görevi gerçek zamanda gerçekleştirmeyi öğrenmektedir. Aynı zamanda, robot da karmaşık ilişkili dinamik sistemleri öğretici insanın yönlendirmeleriyle öğrenmektedir. Bizim araştırmamızdaki temel amacımız, anlatılan sistemde beceri aktarım hızının nasıl artırılacağı ve aktarılan becerinin performansının nasıl yükseltilebileceği sorusuna bir çözüm bulmaktır. Beceri aktarım seansının başında, insan öğretici robotu tamamen teleoperasyonla kontrol etmektedir. Görev performansı geliştikçe, robot kontrolü eline almaya başlamakta ve en sonunda tamamen otonom hareket etmektedir. Önerdiğimiz çerçeve altında gerekli geliştirmeler yapılmış ve sonuçlar hareketli direk(cart-pole) donanımında, hareketli direk denge simülasyonunda ve dağ aracı(mountain car) görevlerinde gösterilmiştir. Eşzamanlı öğrenmenin standart sıralı öğrenme sistemlerine(robotların öğreticiyi gözlediği ve kontrolü eline almadığı öğrenim yöntemi) göre avantajlarını değerlendirmek amacıyla iki gruba bölünen insan öğreticilerle öğrenim deneyleri yapılmıştır. Ayrıca önerilen eşzamanlı çerçeveyi doğrulamak için, insan öğreticiyi modellemek üzere pekiştirmeli öğrenme uygulanmıştır. Elde ettiğimiz sonuçlarda, eşzamanlı öğrenim sürecini tamamlamış ve otonom hale gelmiş kontrolörün belirtilen görevlerde daha yüksek performansa ulaştığı görülmüştür.

# ACKNOWLEDGEMENTS

I would like to thank my advisor Prof. Erhan Oztop for his guidance and support during my master study. I also want to thank all the members of the Ozyegin Robotics Lab members for their friendship. This research was done in the Robotics Laboratory at Ozyegin University;Research supported by European Communitys Seventh Framework Programme FP7/2007-2013 under the grant agreement no. 321700, Converge.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# CHAPTER I

# INTRODUCTION

Human-robot interaction (HRI) field is concentrated to study, design and evaluate robots which are used by human or in a greater extend in human-robot system [2]. Earlier vision pictured robots as substitutes for human. For example, Tesla said "the first of a race of robots, mechanical men which will do the laborious work of the human race "[2]. However, nowadays it is clear that all robotics system have somehow a form of interaction even those seem as full-autonomous [2]. Adaptation, learning and training are one of the factors which affect the HRI system designs. Compared to other issues of the HRI, training has received very little attention [2]. One reason is that the main goal of HRI system design is to have few training. This can be true if the goal is to design specific robots [2]. Therefore, training and adaptation become more important when the robots are multipurpose or universal.

It is generally expected that robots and autonomous agents will become a part of our daily lives in the coming decades. Yet, it is not possible to move robots directly from factories to our homes [2, 3]. One obvious reason is safety [4]; the second reason is that the robots we require in our daily lives will be in charge of a variety of multiple tasks. It is not feasible to program robots in advance for all possible tasks using classical robot programming [3] as they are aimed at making robots specialized to predefined tasks [5]. The classic robot programming methods are applied widely in industry. One of the main drawback of the classic programming is it highly depend on the environment. For example, if a robot is programmed to work in a certain condition and environment, then is necessary to reprogram the robot for new environment. Moreover, even with a fixed environment, changing the target for the robot (e.g.

desired end-effector position of an arm robot) is also force programmers to reprogram the robot. Generally, in a naturalistic setting, robots must be able to automatically self-calibrate, and adapt their -even fixed- behaviors according to the stochastic and dynamic environments they live in [3, 4, 6]. Another important characteristic that sets apart todays industrial robots from the future daily-life robots is the definite need for the ability to interact with humans in the environments designed for humans [4, 7]. Therefore, intuitive and easy robot programming is one of the active research areas in robotics. However, most of the robots in industrial applications have not designed to have interaction with human. Although the aforementioned issues are our concern in future robot programming, as a main problem, all programming procedures are performed by skilled engineers who have significant knowledge of robotics and programming. However, this limits our vision of daily usage of robots in our life. Therefore, more easy and intuitive way of programming is needed. There has been considerable effort in the recent decades to make robot programming intuitive and easy.

This study proposes an intuitive framework for skill transfer which is based simultaneous learning of both human and robot. The framework first allows human to take control of a given task. As (s)he gets successful performance eventually share the control with machine learning methods (learned the demonstrator's policy) to perform the task. And finally, by achieving more success the framework decreases demonstrator's contribution and eventually it become full autonomous. This continuous and performance-based skill transfer is performed via well-defined dynamics equation.

## *1.1 Thesis Contribution*

In the proposed simultaneous learning framework we introduce the novel ideas of (1) state based control sharing, and (2) well defined dynamics for the mixing coefficients

based on overall performance, and implement these on both simulated and physical cart-pole system. The human-in-the loop learning setup employed is similar to that of Ref. [8] where also simultaneous human-robot learning was implemented for balance control. There, however the control was weighted between human and robot simply based on the prediction error of the machine learning algorithm that learns to replicate human actions. In general, weight sharing based on prediction error is not desirable as this may cause the transfer of a bad skill when the demonstrator is consistently performing badly (e.g. doing nothing). This was not an issue in Ref. [8] because the feedback relayed to the demonstrator forced the subject to perform well (otherwise he would fall). Although the work reported in Ref. [8] is in the same spirit of our simultaneous learning, we introduce the novel improvements of 1, 2 mentioned above.

In summary, we define a dynamical system for a weight variable that determines the amount of mixing of robot and human controls. The dynamics gradually change the mixing weight based on the overall success during the progress of the task, which does not directly depend on prediction error of the machine learning module. The dynamics starts off with fully manual (operator control) and tends toward fully autonomous control as overall success level increases. As such, for example, at some point, the system may shift the control towards robot autonomy; but if this results in a reduced success level, then the control will be shifted back towards manual control. Eventually this tug-of-war will enable the robot to gain full control with continued high success in task execution. We hold that the weight dynamics must be state dependent, as some parts of the state space can be learned faster than the others (specially in those tasks that are naturally combination of two or more sub-tasks). In the current implementation, we split the state space in discrete regions, with individual weight sharing dynamics for each region for cart-pole swing-up and balance task. This split can also be viewed as dividing the overall task into manageable subtasks. For experimental evaluation, the developed simultaneous human-in-the-loop

learning framework was implemented on a cart-pole system for the task of *swing-up* and *balance*. The effectiveness of the framework was validated by an expert subject. Good autonomous policies could be generated in very short times (10-30 minutes) compared to sequential learning (see: http://youtu.be/S3NW0hr72mU) . To further assess the effectiveness of the simultaneous learning with naive subjects we made a preliminary experiment with 8 subjects. The subject were randomly divided into two, and assigned to one of the learning setups: the first group engaged in simultaneous learning setup (the robot learned with the subjects and engaged in control, so the robot behavior was not stationary); the second group engaged in standard sequential learning setup (robot behavior was stationary, i.e. the learning did not affect robot behavior). To show the performance of the proposed system clearly, we also only concentrated on the balance task. Therefor, the subjects only needed to focus on a simpler task which have easier control strategy. The results of both experiments indicate that the performance of the final autonomous controller of the simultaneous learning group on average is higher than that of the sequential group. Moreover, using the *Reinforcement Learning* method as a model for human, we evaluate the performance of the system more analytically. This part of study also answers (or gives an intuition) to how the simultaneous learning framework performs better.

## *1.2 Thesis Outline*

Chapter II is about the literature review and introduction of basic concepts. The review part includes different types of robot programming and their basic concepts. Chapter III introduces the framework of human-robot simultaneous learning system. In this chapter, we answer the question of "how" and "when" the skill can transfer from a human demonstrator to a robot. The proposed framework applied on the cart-pole swing-up and balance task which is done by both skilled and naive subjects. In Chapter IV, the reinforcement learning is applied to model human in simultaneous

learning framework. Mountain-car tasked is used to evaluate the performance of the proposed framework. And, finally, we will have the conclusions in chapter V.

# CHAPTER II

# BACKGROUNDS AND RELATED WORKS IN THE LITRATURE

The two mainstream methods in robot programming are: robot learning (self-improvement) [3, 4] and teaching by demonstration (or so called imitation learning) approaches [7, 9, 10, 11] . This chapter focuses more on the literature survey and review of the previous relevant works about the robot programming, robot learning, learning by demonstration and combined learning systems.

## *2.1 Self-Improvement (Reinforcement Learning)*

The basic idea of the self-improvement methods is to find an autonomous controller without human guidance. One of the self-improvement approaches is naturally based on reinforcement learning (RL) [12]. In RL the robot is taken as an agent that explores the state space (which is often the vector of joint angle and angular velocities) to maximize a predefined total (future discounted) reward function such as minimum total cost change to reach a given robot configuration. The RL is neither a Supervised Learning nor Unsupervised Learning. It is not a supervised learning since there is no perfect input-output pair for training and also it is not unsupervised learning since there is sparse feedback which is called reward [12].

The RL has an agent which performs an action in the environment. The current situation of the agent in the environment is called state (figure 1). In other words, state uniquely describes the current situation of the task. The ultimate goal of RL is to find a mapping from state to action which is called policy (equation 1).

$$\Pi(s) = a \tag{1}$$

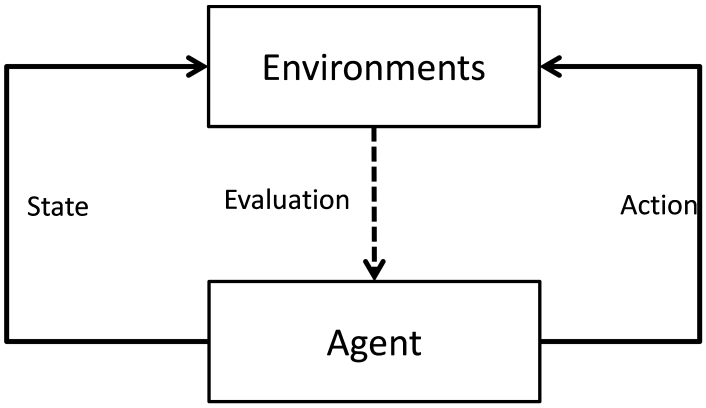


**Figure 1:** General framework of Reinforcement Learning. An agent performs an action based on the current state and gets a feedback or evaluation from environment

Where Π is the policy function, $s$ and $a$ represent the state and action, respectively. The optimal policy can be obtained by maximizing the expected long-term reward. As mentioned earlier, the agent receives a feedback from the environment which is called reward. The reward is generally depends on the previous state of the agent, current state and agents action $R(s^{old}, s^{new}, a) = r$. The reward is a scalar number which describe whether the agent completed the predefined goal. However, maximizing immediate results (greedy policy) will not necessary lead us to complete the task. Therefore, the RL applies maximizing Return or expected long term reward (equation 2).

$$R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + ... \tag{2}$$

Where, R is the return, $r_t$ is the reward, and  is the forgetting factor. The summation continues until the agent reaches to the terminal state. Terminal state or goal state is the state that learning is terminated in episodic tasks and it starts from the initial state. Generally, the RL repeats the following processes [12].

1. Agent observes the state.

2. Agent generates an action based on its own policy.

3. Agent gets the reward from environment.

4. Then it executes the action and goes to the next state.

### 2.1.1 Markov Property

The main reason that why we are allowed to apply the RL to a task is the Markov property. Roughly, when state summarizes the past events in it, then it has Markov property. It means that only current state and action are enough to get the next state and the next reward (i.e. knowing entire history is not necessary) [12].

### 2.1.2 Value Function and Action-Value Function

The value of a state is roughly the expected return starting from that state by following the agents policy (equation 3 ).

$$V^{\pi}(s) = E_{\pi}\left\{R_t | s_t = s\right\} \tag{3}$$

And consequently, action-state value function, which has value for each state-action pair is the expected return starting from the state by taking the action under agents policy (equation 4). Of course, expected returns depend on the policy of the agent.

$$Q^{\pi}(s,a) = E_{\pi}\left\{R_t | s_t = s, a_t = a\right\} \tag{4}$$

### 2.1.3 Q-learning

The Q-learning is one the basic RL methods and well-studied approaches which first suggested by [13]. The Qlearning mainly estimates action-state value function. Straightforward RL is effective for small degree of problems but slow for higher number of degree problems (e.g. $> 6$). With careful selection of hierarchical organization significant improvements can be achieved [14]. Furthermore, there are recent developments that hold promise for fast approximate RL [15, 16, 17]. In learning by

demonstration, the skill of a demonstrator is transferred to the robot [18]. These two approaches are not exclusive and can be used together: the demonstration can be used as an initial rough solution after which RL improves the solution [19].

## *2.2 Teaching by demonstration*

Teaching by demonstration is transfer the skill to a robot using human demonstration.

### 2.2.1 Imitation Learning

Imitation Learning is mainly about on relying on the transfer of the skill to the robot without additional self-improvement (figure 2). So, in this setting the simplest form of skill transfer is to record the demonstrated motion e.g. by motion capture techniques, and play back on the robot to achieve autonomous task execution [10].

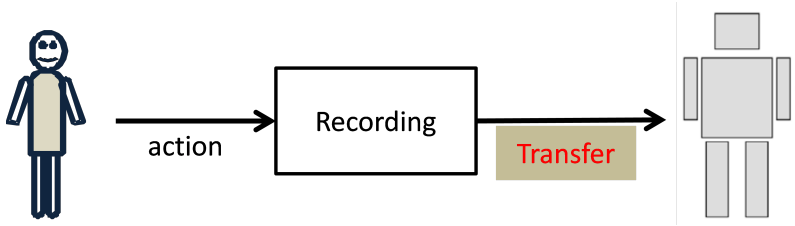


**Figure 2:** Imitation Learning: recording human motion by motion capture system and play back on the robot

However, since the dynamics of robots may change (during learning vs autonomous test). Moreover, it is not suitable for complex tasks as well as those tasks which have non-negligible dynamics. Therefore, certain hand-crafted transformations, and/or inverse kinematics may be needed for this to work as the kinematics of the demonstrator and the robot may be different.

### 2.2.2 Human Guided Direct Teaching

An effective way to teach robots by human guidance is so called *Direct Teaching* [20, 21, 22], where the human demonstrator physically moves the robots joints to

accomplish the given task [23, 24]. Direct teaching is very intuitive; but unfortunately, it is also not suitable for complex tasks that include non-negligible dynamics [20]. The reason for this is the fact the human is not really placed in the control loop of the robot.

### 2.2.3 Imitation Learning with Self-Improvement

Imitation learning with self-improvement is basically similar to the method of imitation learning with a major difference. After recording data, instead of working on the transfer function of the system, the transferred human movement data are applied as an initial point of the RL (figure 3). It means that the RL does not start from scratch and already have good-enough start point.

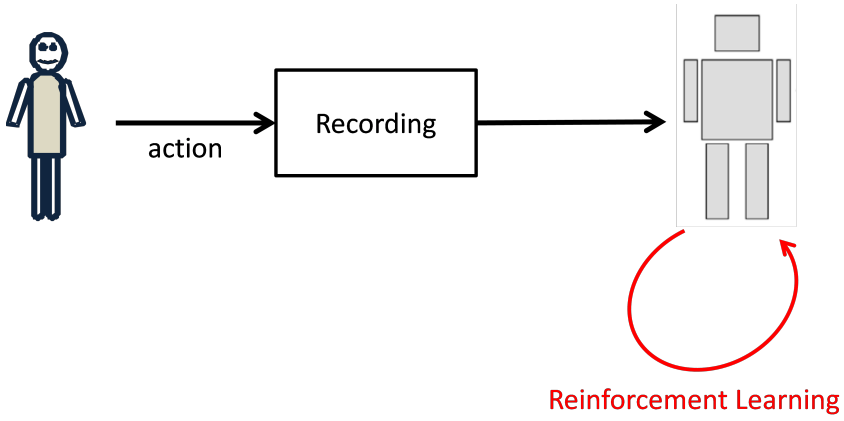


**Figure 3:** The imitation learning with self-improvement

## *2.3 Human-in-the-Loop*

We see two major variants of the human-in-the-loop robot teaching paradigm: In the first variant (sequential learning), the robot is considered as a stationary tool [23, 25] which does not change during the course of human control. This is the type of tasks we experience most in our daily lives, e.g. one's favorite computer mouse behaves the same every time it is used. In the second variant (simultaneous learning), the robot

'learns' together with the human, and this learning is incorporated into the control. So, for the human the task is not stationary anymore. The latter begs to question whether this is really a good idea.

### 2.3.1 Offline, No Control Mixing (Sequential Learning)

In contrast, in the Human Learning for Robot Skill Synthesis paradigm of Babic et al. and Oztop et al. [25, 26] the human operator is placed in the real-time control of the robot as in tele-operation with the ultimate goal of obtaining an autonomous controller based on the performance of the operator (figure 4). In this framework, the human operator/demonstrator must first learn how to do a given task by using the robot as a tool and become skilled at it. This is akin to a beginners learning to drive a car. After human becomes expert at executing the task through the robot, the robot states and corresponding motor commands generated by the human operator are utilized to obtain a policy that drives the robot autonomously [25, 26, 27]. This paradigm has been successfully applied to obtain skills such as ball manipulation [23] with an anthropomorphic robot hand, and balanced inverse kinematics for a humanoid robot [25], and more recently tasks that involve force based policies [24, 8]. It will be fair to say that robotic researchers are becoming increasingly more interested in human-in-the-loop robot control and learning, which provides a platform for both human and robot to learn actively [8, 28, 29, 30].

### 2.3.2 Online, Control Mixing (Simultaneous Learning)

This approach was somewhat used in an ad-hoc way in Ref. [24, 8], however, no study addressed this question directly. In this thesis, we make a contribution in this direction and show the acquisition of autonomous *swing-up* and *pole balance* skill via simultaneous learning. In addition, we present our initial work on comparing sequential and simultaneous learning.

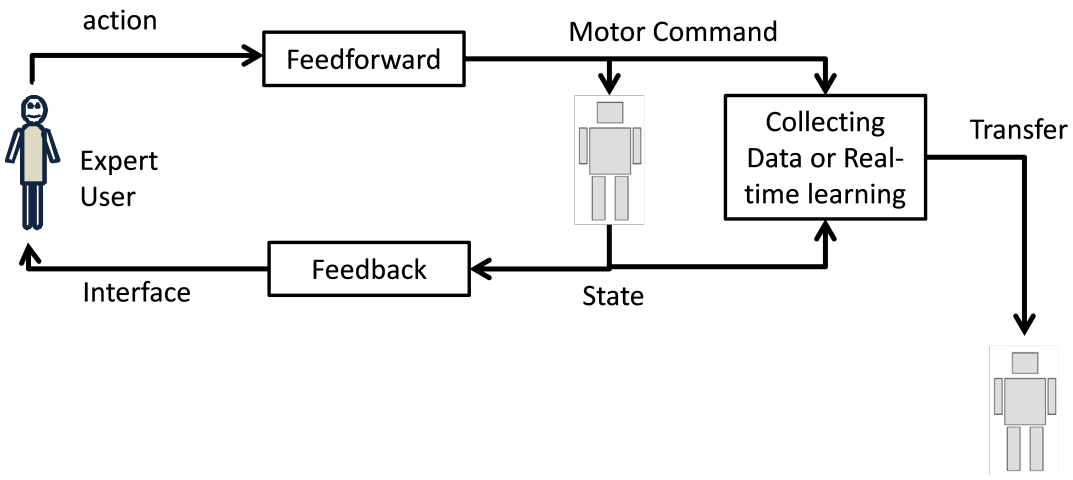


**Figure 4:** Human-in-the-loop with no control mixing

## *2.4 Other Forms of Combined Systems*

### 2.4.1 Human-Human learning system

In the literature, we can see other types of on-line[1] interaction systems in learning a task. One of these cases is human-human learning system. Generally, one the most recent studies about learning with a partner show that they help to improve the performance in the task. Reference [31] set an experiment to evaluate usefulness of learning with partner. In this experiment, two human performed a point tracking task on a screen which was moving randomly. Both of them could not see their own arm position and also the arm of the partner. Although they did not have any physical contact with each other, they arms were connected by a virtual band. That means they arm are pulled toward the position of the partner's arm. Moreover, most of them also were not aware of working with a partner. Regardless of the partner's skill and performance, they could consistently improve their performance in a dual team. Furthermore, the results of dual performance were better than those who practice alone. This study also find partner in same level get more benefit of dual learning

[1]On-line here means they can feel the force of other pair during learning either physically or by change in their contribution of control.

rather than working with an expert. The subjects get better results when they work with a human than a robot.

# CHAPTER III

# CONTROL MIXING AND SIMULTANEOUS ADAPTATION FOR POLICY TRANSFER

## *3.1 Human-Robot Simultaneous Learning Framework (Cart-Pole Swing Up and Balance in Hardware)*

We propose a framework that aims to engage human and robot learning simultaneously to improve the effectiveness of human to robot skill transfer. This may be seen analogous to the learning process between a human teacher and a human learner [8]. The learner tries to mimic the teacher's action gradually by getting help from the teacher when needed. As the learner becomes better at satisfying the tasks objectives, the teacher involves less. However, when this deteriorates performance the teacher intervenes back and issues corrective actions. In this framework, likewise, the demonstrator has the full control of the task in the very beginning, and gradually the control is shifted to the robot based on the overall performance, so if the performance degrades the human demonstrator's share in control increases. When the success objectives are completely satisfied, the robot becomes the only controller. At this point, the learned policy can be preserved as it is and deployed later for full autonomous execution of the task without human guidance.

Figure 5 presents our human-in-the-loop simultaneous learning framework. Human demonstrator controls the robot in real-time via fixed feed-forward interface, and observes the robot state through a feedback interface (which could be simply direct vision). At the same time, the robot starts to mimic the human policy and injects its own control gated through a weight parameter. The final action (u) is simply the linear combination (equation 5) of the human and machine action ($u_h$,$u_m$

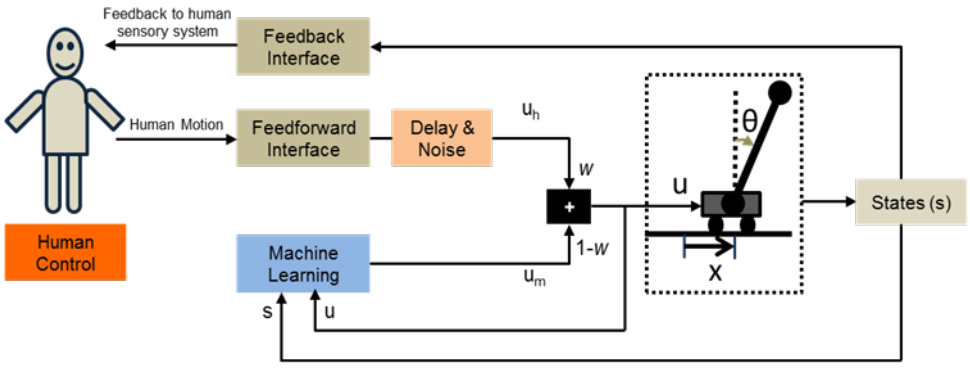


**Figure 5:** Simultaneous human-in-the-loop robot learning framework is illustrated. Human demonstrator controls the robot in real-time to achieve a desired goal through the robot . Simultaneously, the robot learns to imitate the human policy. The net control the robot receives is obtained by the weighted summation of the human and machine generated controls. The weighting is dynamically adjusted so as to pass the control to the robot when successful learning and task completion is achieved. At this point the task skill has been transferred to the robot and it can complete the task autonomously without requiring human guidance.

respectively):

$$u = w_h + (1 - w)u_m \tag{5}$$

Where $0 \leq w \leq 1$ indicates weight or share of the human action on the overall control. The key design issue is to define a $w$ dynamics to facilitate effective skill transfer avoiding oscillations between human and machine control. In this proposed method, the mixing weight value neither is controlled by human nor robot. An independent algorithm based on the performance of the human-robot system decides to tune the mixing weight. For this we propose to use a time-windowed success indicator to determine the dynamics of w. The proposed framework does not specify the details of the involved feedback or feed-forward interfaces. In the current implementation direct vision is used as the feedback. For the feed-forward interface we used a standard computer mouse: the horizontal velocity were multiplied with an experimentally tuned constant gain to obtain the voltage that drives the cart of the cart-pole system.

### 3.1.1 Cart-Pole swing up and balance task

We chose a cart-pole system to test the proposed simultaneous human-in-the-loop learning framework (see figure 6) . The desired controller would move the cart left and right several times to accumulate sufficient energy to swing up the pole and then keep it at the vertical upright position. So, the desired state manifold is given by $\theta = \dot{\theta} = 0$. The control policy is captured by a time independent function of the cart-pole state u=g(X). The goal of the human-in-the-loop robot learning is to have the human produce control data points that can be learned by a machine learning algorithm so that g is obtained and can be used as an autonomous controller for the task. Here u is the controller output (voltage). X is the cart-pole state which is normally defined as $X = [x, \dot{x}, \theta, \dot{\theta}]$ where $x$ is the position of the cart, $\dot{x}$ is the velocity of the cart, $\theta$ is the angle of the pole, $\dot{\theta}$ is the angular velocity of the pole. Alternatively, the state definition can be chosen $X = [x, \dot{x}, \cos\theta, \sin\theta, \dot{\theta}]$ which avoids the discontinuities due the periodic nature of $\theta$, and so make the task of machine learning module easier.

### 3.1.2 Supervised Learning

An important notion of human-in-the-loop skill synthesis is that data is the key not the underlying machine learning algorithm, as the framework allows generation of large data sets as the human is placed in the control loop [26, 27]. So we do not focus on the specifics of the machine learning algorithm to represent g(), but instead underline the requirements for a generic machine learning algorithm for simultaneous learning: (1) support for online incremental learning, (2) robustness against over-fitting, and (3) low computational load. A good match for these requirements is the Receptive Field Weighted Regression (RFWR)[1] algorithm [32].

[1] http://www-clmc.usc.edu/Resources/Software

**Figure 6:** The experimental setup of the cart-pole system.

### 3.1.3 Weight Dynamics

One of the critical part of the learning is how fast to pass the control from the demonstrator to the robot. If the robot takes the control too early it may not learn the task properly, and it may also hinder the corrective actions that the demonstrator may take. Obviously, if the non-stationarity introduced is too severe (e.g. by a fast change of the weight), it may prohibitively increase the task complexity for the human demonstrator. On the other hand, slow transition of the control can be frustrating for the demonstrator. The aforementioned factor can be reflected as a time constant which presents how fast the machine can take the control during the demonstration.

Another factor is the acceptable level of task performance that should indicate a shift towards autonomy (e.g. shall we expect the demonstrator to find the theoretical optimal policy?). So it is reasonable to introduce a success measure that (time-wise) locally captures the performance of the human-robot system, and use it to guide the weight dynamics. As the weight parameter $0 \leq w \leq 1$ determines the level of

autonomy, we arbitrarily define that *w=1* to mean full human (manual) control, and *w=0* to mean full robot (autonomous) control. As it is motivated above, we wish to have the control passed to the robot when the task performance is high. This intuition can be captured with the following w dynamics (equation 6):

$$\tau \dot{w}(t) = -w(t) + f(success(t)), w(0) = 1 \tag{6}$$

Here, $\tau$ is the time constant, $0 \leq success \leq 1$ is the measure for the temporally local task performance, and f is fixed monotonic function of success bounded by 0 and 1. If *f=0* $w$ will settle to zero, meaning that the control has been completely passed to robot. So, intuitively *f(success)* should indicate the need for human guidance. There are infinitely many possible $f$ choices; in the current implementation, we selected the simplest without further investigating the effect of this choice.

$$f(success) = 1 - success \tag{7}$$

With this the dynamics of w becomes simply

$$\tau \dot{w}(t) = -w(t) + 1 - success(t), w(0) = 1 \tag{8}$$

### 3.1.4 Success Measure and State Space Partitioning

In the cart-pole system an intuitive success measure is the (average) height of the tip of the pole. The goal in our cart-pole task is to bring the pole from downward hanging posture to the upright vertical posture and keep it there as long as possible, where the height achieves its maximum. Representing the angle of the pole from the vertical upright posture with $\theta$ we can take $\cos\theta$ as the indicator for the height of the pole tip as a value in [-1, 1] irrespective of the pole length. From now on, we will use the term pseudo-height to refer this value.

It is reasonable to think that the success should be defined based on state or state

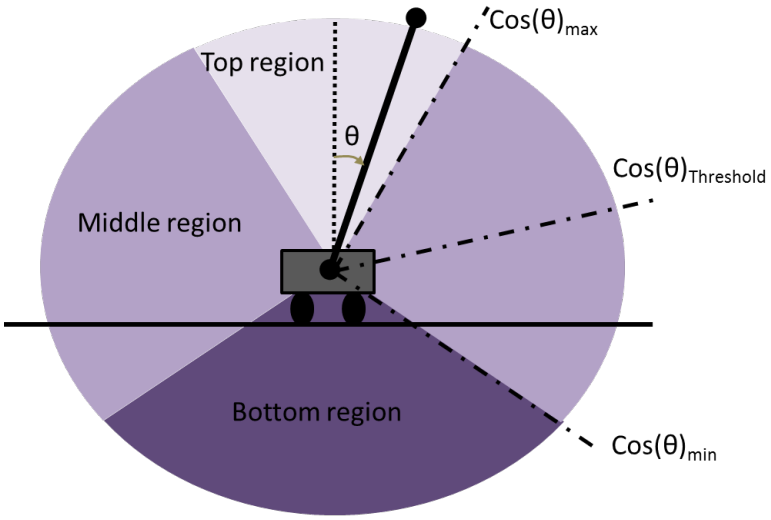


**Figure 7:** State Space Portioning of the cart-pole problem.

regions. For example, one may be good at driving at low speeds; but, may be terrible at high speeds. This naturally brings the idea that a task can be divided into subtasks via partitioning of its state space. This is in fact a common practice in (hierarchical) reinforcement learning [14]. It is not uncommon to see swing-up and pole balancing as different benchmarking tasks in the literature. Moreover, teleportation system need full direct attention of human. Therefore, by splitting the task into subtasks there are chance to make the condition more relax for user. And, eventually human become supervisor than demonstrator. By generalizing from this, we defined three regions in the pole angle space (see figure 7): *bottom* region (start), *middle* region (swing-up) and *top* region (balance). And, similarly we defined two regions for angular velocity: *slow* and *fast*. With this, we partitioned the state space into 2x3=6 regions, for which we defined individual success measures as:

$$success_i(t) = h(\frac{\overline{cos\theta_i(t)}^{t}_{t-\alpha} - cos\theta_i^{min}}{cos\theta_i^{threshold} - cos\theta_i^{\min}}) \tag{9}$$

where $h$ is given by

$$h(x) = \begin{cases} 0, & \text{if } x < 0 \\ x, & \text{if } 0 \leq x \leq 1 \\ 0, & \text{if } x > 1 \end{cases} \tag{10}$$

and $cos\theta_i^{\min}$ indicates the minimum pseudo-height of the pole tip in region $i$, $cos\theta_i^{threshold}$ is a constant threshold set for region $i$, and finally $\overline{cos\theta_i(t)}_{t-\alpha}^{t}$ is the moving average of the pseduo-height within the past $\alpha$ time unit. For convenience the state regions are named as down-slow, down-fast, middle-slow, middle -fast, top-slow and top-fast. Since we partitioned the state space based on the height and angular velocity of the pole, each state region is determined by four parameters of of $cos\theta_i^{\min}$, $cos\theta_i^{\max}$, $|\dot{\theta}|_i^{\min}$ , $|\dot{\theta}|_i^{\max}$. The parameter $cos\theta_i^{threshold}$ determines the performance level required for that region to be considered successful. We selected the threshold relatively lower for the swing up task and higher for the balance task. This settings helps the demonstrator to clear the swing-up task faster and have it handled by the robot for focusing more on the balance task. The aforementioned parameters are given in Table 1. Since the success is defined to be state dependent, the control sharing also becomes state dependent. Hence the dynamical equations governing the weights for each region is given by:

$$\tau \dot{w}_i(t) = -w_i(t) + 1 - success_i(t), w_i(0) = 1 \tag{11}$$

### 3.1.5 Obtaining the Autonomous Controller

Since each state region has independent learning dynamics, the (sub)task completion (i.e. *w=0* and *success=1* for the particular region) may be attained at different times for each region. The overall skill is already transferred to the robot, when the weights for each region approximately reach 0, as at this time the human contribution to control would be minimal. At this point the parameters of the machine learning

**Table 1:** The cart-pole state region parameters. The average achieved height obtained by subjects in a time window is compared with the threshold value

| | $cos\theta_i^{\min}$ | $cos\theta_i^{\max}$ | $\lvert\dot{\theta}\rvert_i^{\min}$ | $\lvert\dot{\theta}\rvert_i^{\max}$ | $cos\theta_i^{threshold}$ |
|---|---|---|---|---|---|
| *Down Slow* | cos(180°) | cos(120°) | 0 rad/s | 8 rad/s | cos(160°) |
| *Down Fast* | cos(180°) | cos(120°) | 8 rad/s | 20 rad/s | cos(160°) |
| *Middle Slow* | cos(120°) | cos(30°) | 0 rad/s | 3 rad/s | cos(70°) |
| *Middle Fast* | cos(120°) | cos(40°) | 3 rad/s | 20 rad/s | cos(60°) |
| *Top Slow* | cos(30°) | cos(0°) | 0 rad/s | 1 rad/s | cos(4°) |
| *Top Fast* | cos(40°) | cos(0°) | 1 rad/s | 20 rad/s | cos(20°) |

system is retrieved and saved as the autonomous policy that can be deployed at a later time. There is a practical issue that must be handled to make the life of the demonstrator easier: if the demonstrator fails to continue to provide appropriate action (e.g. due to tiredness or lack of attention) for the state regions that has been taught to the robot, success will decrease and the control weight will be shifted back to the human. In other words, thinking that a state region is learned by the robot and thus can be conveniently taken care of the robot will not really work in the standard form. However, it is easy to gain this convenience by freezing the learning for regions that attain high level autonomy during the overall training period. In the experiments reported this strategy was adopted.

### 3.1.6 Results

The proposed simultaneous human-robot learning framework for robot skill transfer was evaluated on a physical cart-pole system. The experimental sessions started with the pole positioned downward with zero velocity and the cart placed at the center of the cart-track. The subjects were asked to swing up the pole and keep it there in balance. There were no instructions for the position of the cart. The system could be operated in manual (sequential mode) where the sole control is given by the human, and data collection was performed for subsequent machine learning; or in simultaneous

mode, where the proposed learning and control sharing was employed. The open parameters for the proposed framework are the time constant weight dynamics ($\tau$) and the size of the moving average window ($\alpha$) that is using computing the task success (success parameter). Both parameters were chosen as 40 for the experiments reported in this paper.

The initial experiments were done with a subject who had extensive experience with the experimental setup, and thus could do swing-up quite easily but could not keep the pole in upright position for more than a few second via manual control. When the expert subject was asked to perform the task in simultaneous mode he was able to perform better in pole balancing[2]. In fact, the autonomous controller obtained as the result of 30 minutes of simultaneous learning could swing up and hold the pole in the upright position. During the experiment, the expert first completed the regions which are likely to be involved in the swing-up (down-slow, down-fast and mid-slow) which were easy to complete. Hence the weights of these regions converge to fully automatic ($w \approx 0$) much faster (see figures 8 and 9 dashed lines). On the other hand the state regions that were more involved in pole balancing (top-slow, top-fast, and mid-fast regions) took longer to reach $w \approx 0$ level (see figures 8 and 9 solid lines).

A set of additional experiments with 8 naive subjects were carried out to assess the validity of the proposed method for naive subjects, and compare it against the sequential learning scheme. The subjects were randomly divided into two groups of 4. All subjects were allowed to work with cart-pole system for 3 minutes to familiarize themselves with the human-in-the loop cart-pole control setup. Then, after a short break, they were given the robot control task for 7 minutes. First group learned simultaneously with the robot, and the second group performed the task based on the sequential learning scheme, where robot did not change its behavior during human

[2]The video of a learning session that and autonomous performance can be viewed at http://youtu.be/S3NW0hr72mU

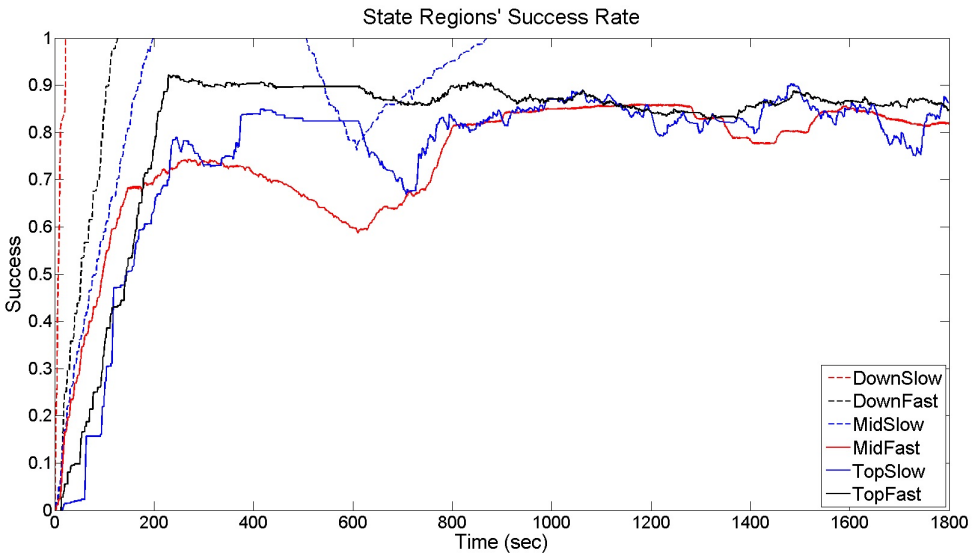


**Figure 8:** Success level of a skilled demonstrator in each state regions. 100 % Success level reached in early time of easy-to-learn state regions. In the rest of the graph,the level oscillates between 70% and 100% for Middle Fast, Top Slow, and Top Fast state regions. (see http://youtu.be/S3NW0hr72mU)

learning. Average pseudo-height during the first thirty seconds is used to assess the performance of the autonomous policies obtained via the sequential and simultaneous learning[3] . In addition, the performance during the human control of the robot is also calculated. This corresponded to simple teleoperation in sequential mode and 'combined control' (human and machine) in the simultaneous mode. For testing the autonomous performance the pole was initialized to the downward vertical position and given a small fixed perturbation (this was necessary to get some 'poor' policies[4] started, otherwise we would not be able to make within 'poor policy' comparisons). The simultaneous learning based autonomous policies performed better in average (-0.50) than the sequential mode based policies (-0.65) (see Table 2). However the difference was not statistically significant as can be understood from the standard deviation given in the table. To gain an understanding at the individual level, we

[3]This duration of 30 seconds was found to capture the full behavior of the autonomous controllers obtained; however it is not critical and could have been chosen longer.

[4]By poor policy we refer to those autonomous policies which could not start swing-up task.

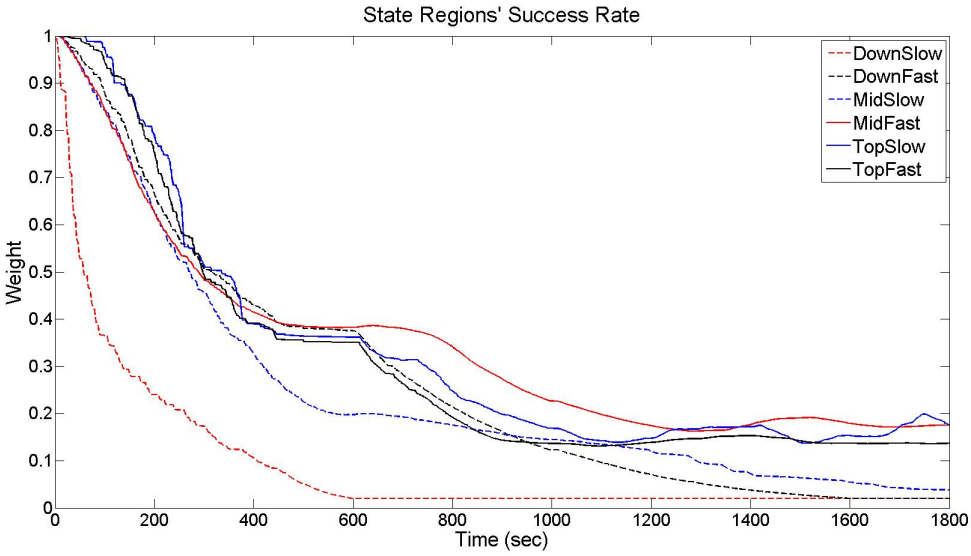


**Figure 9:** Control sharing weight in each state region for a skilled demonstrator is shown during simultaneous learning. Weight starts from 1 (fully manual) and approaches to 0 (autonomous), gradually transferring the skill to the robot. This indicates different state regions have different difficulties.

sorted the subjects in each experimental group according to their performance and generated the performance vs. subject order plots (figure 10). The worst subject of simultaneous group generated a policy worse than all sequential subjects[5] . Except this, simultaneous learning seems to generate better autonomous policies quantified as average pseudo-height as can be seen from figure 10. An interesting observation is that the difference between the best and worst sequential policies is so small, which can be seen from the standard deviation in Table 2. This is very different in the simultaneous learning based policies where the best subject performance is really good. The generated policy from him could do swing up, and also pole balancing to some level. The performance and performance variation seen in autonomous policies could be also seen during learning (see Table 2, Human Control vs. Human + Robot Control).

[5]This poor performance might be an outlier as the zero-control performance level (Fig 5, dashed horizontal line) is very close to his performance.

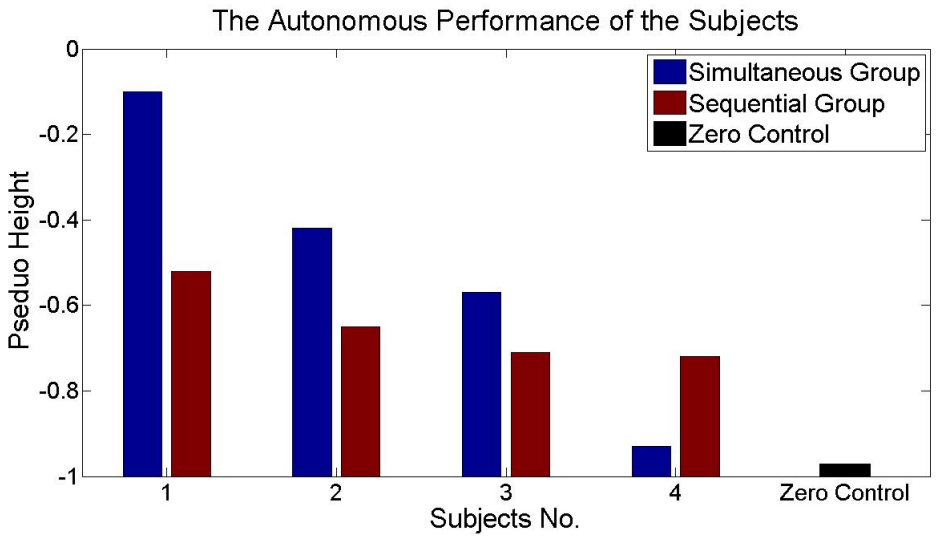


**Figure 10:** Average pseudo-height obtained from the autonomous policies obtained from the subjects in simultaneous (red) and sequential (blue) experimental conditions. The pole was initialized to downward vertical and a small perturbation was applied. The horizontal dashed line indicates the zero control (i.e. the voltage to the cart pole system was 0 V) performance of the perturbation.

## *3.2 Simultaneous Learning for Balance Task in Simulation*

Swing up and balance task is extremely hard for a naive subject. Therefore, to understand the efficiency of the algorithm, the task is focused only on the balance task. For balance, it is necessary to release the pole from the top point. Implementation of cart-pole balance on hardware needs some extra hacks. First idea is a release mechanism which will hold the pole at the top and then release it. This needs an extra device which adds unnecessary complexity to our research and examiner intervention that will increase the experiment's time. The second idea is to perform a swing up and balance (by a classic) controller and then release it. This approach has also a problem. Because, it wastes time to bring it into upright position. So, a simulation application is developed to evaluate the simultaneous learning. This software helps us to test it anywhere without having expensive lab setups.

**Table 2:** The experiments results taken from naive subjects. The results presented as average pseudo-height over time of experiment or autonomous test. HC=Human Control, ARC=Autonomous Robot Control, HRC=Human+Robot Control Seq=Sequential and Sim=Simultaneous.

| Experiment Type | HC (Seq) | ARC (Seq) | HRC (Sim) | ARC (Sim) |
|---|---|---|---|---|
| *Mean* | -0.42 | -0.65 | -0.37 | -0.5 |
| *STD* | 0.1 | 0.09 | 0.23 | 0.35 |

### 3.2.1 Cart Pole Model

Cart-pole has 4 states. Despite of previous part that we used $\sin\theta$ and $\cos\theta$ to cover dis-continuity in the $\theta$ space, $\theta$ is continuous in the balance task ($-30° < \theta < 30°$). Notations and direction can be in figure 11 .

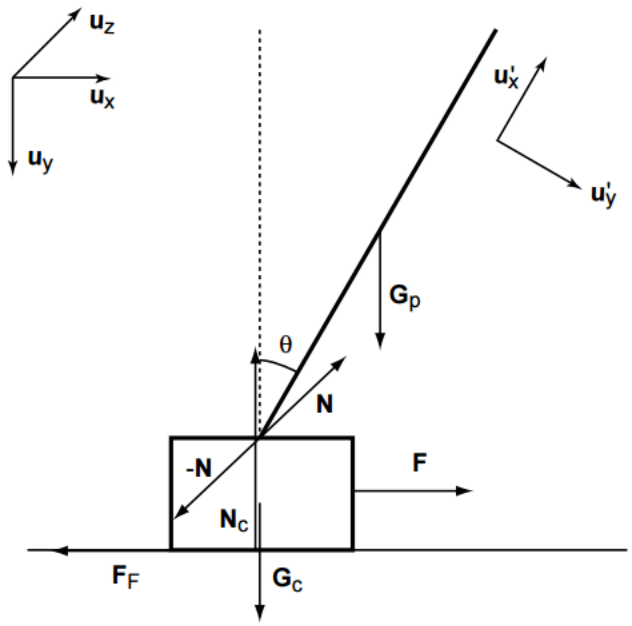


**Figure 11:** Cart-pole system [1]

To implement real-time simulation, differential equation of cart pole applied. Since, estimation of cart and pole friction coefficient is not in the scope of this research, temporary we assume there is no friction. If the friction be neglected, the

equations are [1]:

$$\ddot{\theta} = \frac{g \sin\theta + \cos\theta(\frac{-F - m_p l \dot{\theta}^2 \sin\theta}{m_p + m_c})}{l(\frac{4}{3} - \frac{m_p \cos^2\theta}{m_c + m_p})} \tag{12}$$

$$\ddot{x} = \frac{F + m_p l(\dot{\theta}^2 \sin\theta - \theta \ddot{\cos}\theta)}{m_p + m_c} \tag{13}$$

However, to make it more realistic, we manually add friction to the above equations. After calculating $\ddot{\theta}$ and $\ddot{x}$, the equations are modified by equations 14 and 15.

$$\ddot{\theta} = \ddot{\theta} - (\sigma_\theta + \rho_\theta)\dot{\theta} \tag{14}$$

$$\ddot{x} = \ddot{x} - (\sigma_x + \rho_x)\dot{x} \tag{15}$$

The $\sigma_\theta$ and $\sigma_x$ are constant variables and $\rho_\theta$ and $\rho_x$ are small random variables. These values are obtained by try and error to be similar to the hardware cart-pole. The cart-pole track is limited to 1m and in the two side of the track there is an obstacle. If the cart hit the corner it will bounce back a little. We used two piece of Styrofoam to prevent cart from damage. To model that we simply change the direction of the speed, and also limit the $x$ between $\min(x)$ and $\max(x)$. Therefore, if the cart hit the corner the speed changes to

$$\dot{x} = -\epsilon\dot{x} \tag{16}$$

Where $\epsilon$ is small coefficient got by try and error. It should be noted that it is NOT important here that our simulations become exactly similar to the real hardware. Because, anyway it become more or less a task to be learned by subjects. And the scope of this research is about the learning, not modeling. However, we made our best to dynamically become similar to the real hardware.

To realize the simulation in real-time, we applied Euler integration. So, in each time step, after calculation of the $\theta$ and $x$ and consideration of the frictions, Euler integration is employed in the following equations:

$$\begin{cases} \dot{x}(k) = \ddot{x}(k).T + \dot{x}(k-1) \\ x(k) = \dot{x}(k).T + x(k-1) \\ \dot{\theta}(k) = \ddot{\theta}(k).T + \dot{\theta}(k-1) \\ \theta(k) = \dot{\theta}(k).T + \theta(k-1) \end{cases} \tag{17}$$

Where, $T$ is the time step (which was 20ms here), and k refers to the discrete time. Since by 20ms the simulation may end up with poor integration, we internally integrated 10 times more in each time step.

### 3.2.2 Parameters and Experiment Description

We apply same framework for human-robot skill adaptation. The pole is released from the top with random initial angle between -5 deg and 5 deg. The goal is to keep the pole as much as possible between -30 and 30 deg. If pole passes these angles, the episode is terminated and a new rollout starts. The maximum time in each episode is 20 secs which means the subjects successfully accomplished the task's objectives. The main criterion to evaluate the subject performance is the average keeping time over 100 episodes. Similar to the swing up experiment, mixing weight changes based on the success measurement which is a moving average reward. The reward is calculated in every time step. Since keeping pole at the top is more desirable, the reward equation is:

$$r(k) = \frac{\theta_{\max} - |\theta(t)|}{\theta_{\max}} \tag{18}$$

Where, $|\theta(k)|$ is the absolute value of the angle, $\theta_{\max}$ is the maximum angle which by passing it the episode is terminated (here, $\theta_{\max} = 30^{\circ}$) $r(k)$ is the reward in each

time step ($k$). If the demonstrator can keep it in $\theta = 0^\circ$ position, the reward is 1. And if the $\theta = \theta_{\max}$ the reward is 0. We could not use the pseudo-height for pole balance task, because at the top $cos\theta$ does not have good enough resolution around $0^\circ$. And, we could not also use time since we only measure the time after the episode finishes which makes us to change our learning dynamics to discrete which is not desirable. The success measurement is defined in equation 19 :

$$Success(k) = \frac{\sum_{i=k-\alpha}^{k} r(i)}{\alpha} \tag{19}$$

Where, $\alpha$ indicates window size of the success measurement. In this experiment $\alpha$ is selected as 4000 time step. This number obtained by try and error. Since the frequency of our simulation is 50 Hz, it means that the success measured over past 80 seconds. Similar to the previous experiment mixing weight dynamics is:

$$\tau \dot{w}(t) = -w(t) + f(success(t)), w(0) = 1 \tag{20}$$

Because different subjects have different learning curve, in balance experiment we addressed this issue in an adaptive form of $\tau$. The learning framework system keeps the mixing weight as 1 until the success reach a certain level (here, we select 40%). As can be seen from equation 20 , we proposes an exponential curve for mixing weight. On the other hand, in an ideal case (Success=1), the solution for $w$ is

$$w(k) = e^{-\frac{k}{\tau}} \tag{21}$$

As we mentioned earlier w follows the $1 - Success$, intuitively, an approach to find respect to the subjects'performance can be as following:

$$0.4 = e^{-\frac{k_{Success=0.4}}{\tau}} \tag{22}$$

The time which takes the subject reach to 40 % success is chosen as a reference point. Therefore, the $\tau$ is:

$$\tau = \frac{-k_{Success=0.4}}{\ln 0.4} \tag{23}$$

Another main difference is that in this part there is only one state region. In fact, top-slow was one of the 6 state regions of the cart-pole swing up and balance task. Moreover, since pole is released from top point (around zero angle) without initial angular velocity, therefore, it cannot reach a fast angular velocity like the previous experiment (which makes one states region enough to implement the pole balance task). The computer mouse cursor's vertical speed are converted as the cart-pole force. In previous experiment, the voltage are selected as the user control signal, in simulations force is chosen because dynamics equation are based on the force.

### 3.2.3 Graphical User Interface

A user interface is developed to take the experiments in simulation (figure 12). There are variety of options allows to take experiment easily from subjects. This Graphical User Interface (GUI) is design to do 4 experiments (*Cart-Pole Swing up* and *Cart-Pole Balance* is operational in the time of writing this thesis). As mentioned earlier, three type of experiments are *introduction* (subject becomes familiar with experiment), *Main experiments* (subject plays for a certain amount of time or episodes to teach the robot), and *Autonomous* (using obtained subject's autonomous controller to perform the task). *Simultaneous* and *Sequential* are two categories of the experiments that are studied in this thesis. *LWPR* and *RFWR* are applied as the online learning method. After these settings, user can enter a subject's name which all files and folder created with this name. Since, the examiner may want to continue the experiments for more trials, a number is assigned to each trial of the subject. There are option for the duration of the experiment (for swing up task), and number of rollout for

balance. The examiner can change the length of the pole. It especially helps the subject in the balance task. Intuitively, it is easier to keep a longer pole balanced on your hand rather than the shorter one. The default value is 1m length for the pole. The GUI benefits from some graphical options may psychologically enhance the performance. *Hiding the mouse pointer* (which may distract the subject, when the feedforward interface is the mouse velocity), *Popup graphics* (Open the task graphics in a new figure without any control panel options), *Show Progress* (for showing mixing weight and success during the simultaneous experiment). Two option for controlling by computer's mouse is provided. They can either control by the cursor's vertical velocity or cursor's vertical position. In case of velocity, simply the velocity is fed as the force to the cart-pole simulation. In the position case, it is necessary to generate proper force to match the cart-pole position with the cursor position on the monitor. Between these two options, velocity is more proper. Because, the user may want to immediately change the position (by changing the cursor's position) which is not physically possible. Therefore, they may feel that there is a delay in the system. To adjust the mouse speed, there is a tuning the bar (*Mouse Sensitivity*) that allows users to find a desired sensitivity. There also some reserved areas which show the performance and the saved file names. In the balance task, subjects can see the pole's position for at least 3 seconds in the beginning of each episode. Meanwhile, there is a yellow bar which demonstrates the count-down for releasing the pole, and it prevent subjects to be surprised (figure 13).

### 3.2.4 Expert Subject Results

An expert subject is asked to perform the task the balance task. The performance is evaluated in simultaneous and sequential experiment. Although the expert demonstrator already learned the task, it verifies the advantage of the simultaneous learning that leads to a higher performance.

The results are presented in two parts. First part is related the performance of the expert user when performing the task. And, the second one is the performance of the autonomous policy obtained from the expert which is much more important.

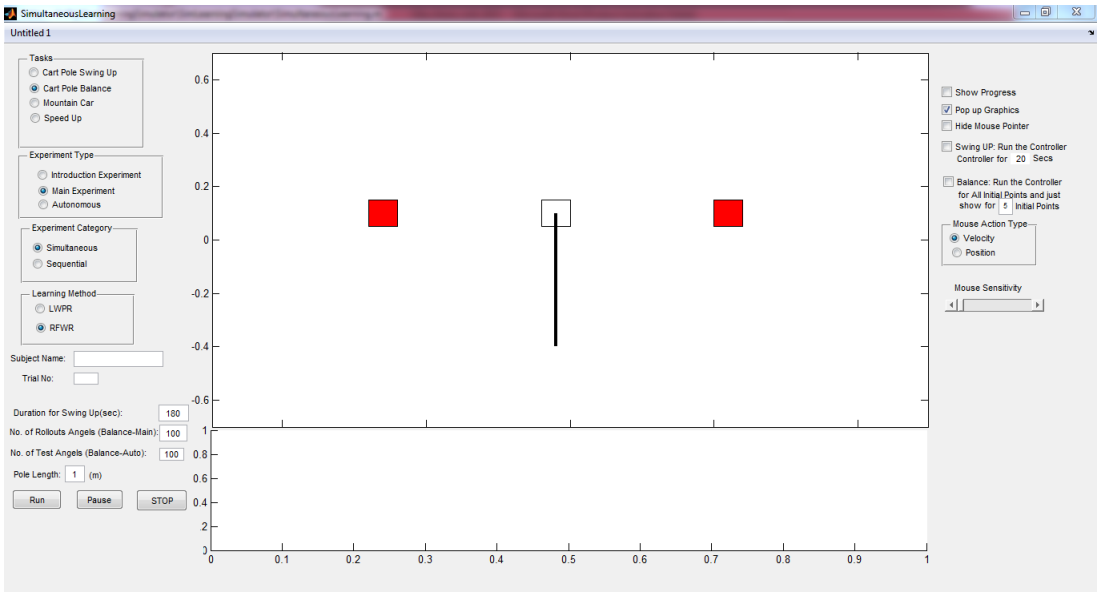


**Figure 12:** Graphical User Interface (GUI) of the cart-pole system.

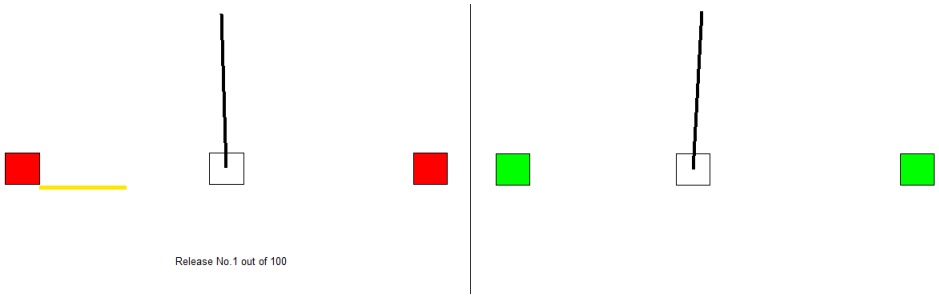


**Figure 13:** Simulation of cart pole setup. The left image shows the waiting situation. A yellow bar will inform the user how long it will take to next release. The right image shows the cart pole during the training. There are two square which shows the limitation of the track and also their color demonstrates the user's responsibility. Red means wait and green means allowed to move.

#### *3.2.4.1 While Learning Performance*

The expert user performed the balance task in 100 rollout which the pole randomly released from [-5, 5] degree. The expert balance time is presented in figure 14.

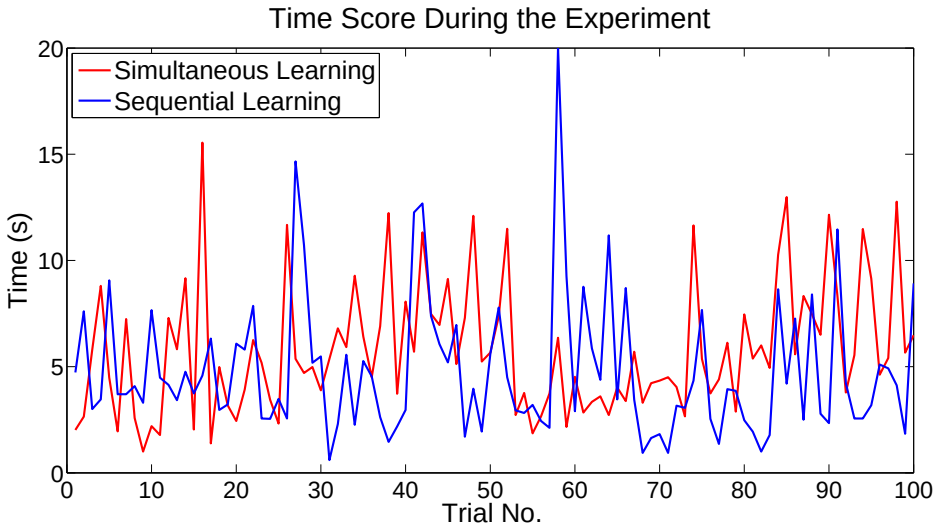


**Figure 14:** The time of balance in each trial.

To make it clear to see the trend in the performance, a simple moving average method is applied with a window length of 10. The results is demonstrated in figure 15. As can be seen, the expert start both simultaneous and sequential experiments with almost same performance. It should be noted that both simultaneous and sequential experiments have same condition in first 20 trials (This number is approximate and depends the subjects performances.). However, at the end of the experiment the expert can hold the pole longer in simultaneous group with considerable gap. It is around 5 seconds (this average of the last 10 trials) and 7.5 seconds in the simultaneous group. To understand how the dynamics of weight mixing works, the figure 16 represents the success measurement of the expert demonstrator.

Mixing weight which is corresponding to 1-success is illustrated in figure 17. The mixing weight is forced to be 1 at the beginning of the simultaneous experiment and after the success reach to 40% then it follows the mixing weight dynamics equation.

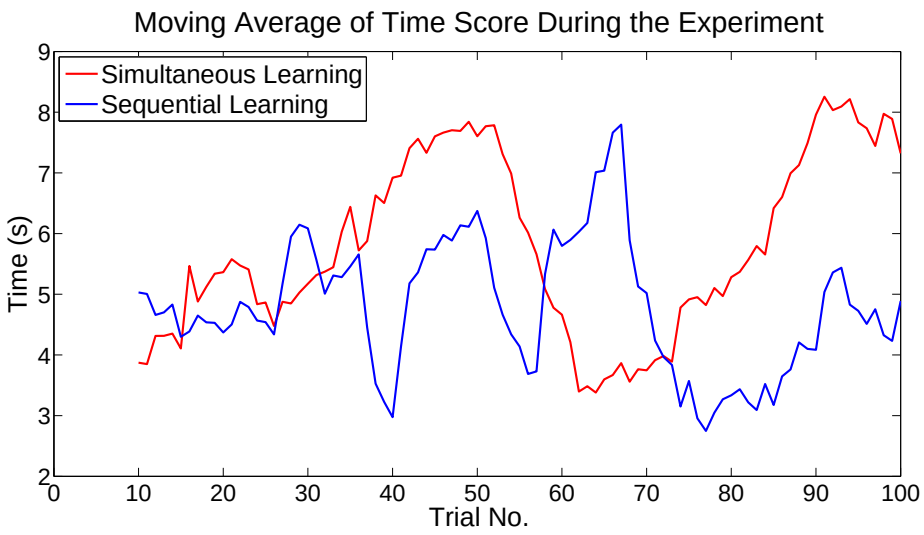


**Figure 15:** The moving average of balance time in each trial.

There are three reasons to start the dynamics equation after that success threshold. First, it prevents change in the system at the beginning which the subjects are naive. The second one is that the demonstrator's performance can be compared with the sequential one (because, it is same at the beginning). And, the third reason is that the applied machine learning algorithm needs enough data to generate reliable actions. To analyze the performance of the expert, the time of balance is presented based on the release angle (see figure 18, figure 19). Figure 18 and 19 include the average of the time for each initial angles, as well as, the average around each angle with 9 neighbor points (red line), total average (black dash line), and zero control (blue curve). Zero control, here, means that the force is constantly zero during the experiment. As can be seen, around zero it take more time for the pole to fall. Figure 18 shows that the demonstrator performance is not highly biased by release angle in sequential experiment. However, the expert could keep the pole longer in angles around zero.

The two performances for the expert demonstrator are also compared in figure 20. The results show that the expert had higher performance on average during the simultaneous experiment.

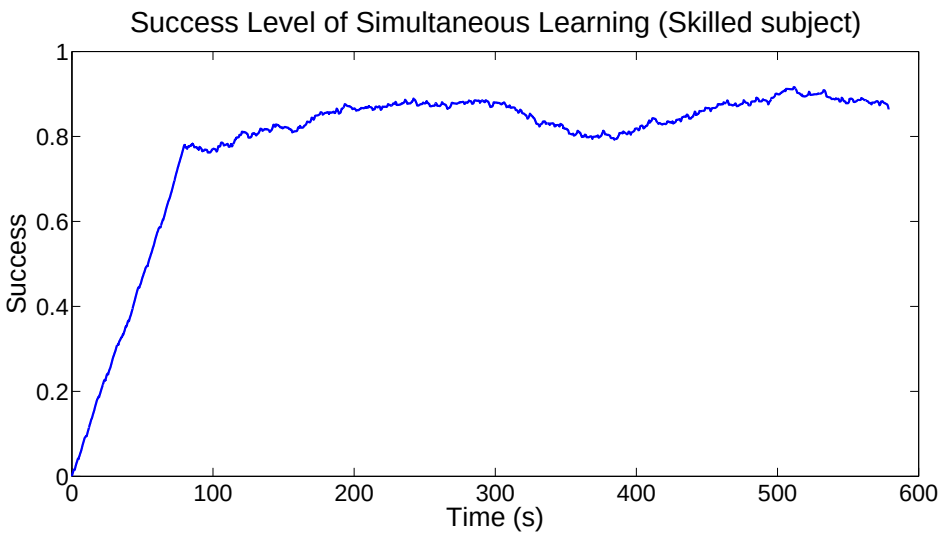


**Figure 16:** The Success level of the expert user.

#### *3.2.4.2 Autonomous Controller Performance*

When the training finishes, the system obtains a policy from the demonstrator (in both simultaneous and sequential experiment). The policy which also called autonomous controller in this thesis can perform the task without human intervention .Of course, the performance of the obtained policy is directly related to how it is trained. The autonomous controller is evaluated over 100 different initial points. The results are depicted in figure 21 and 22. To illustrate the difference much more clear, figure 23 shows the comparison between simultaneous and sequential performance. The controller obtained from simultaneous training can keep the pole longer than the sequential one (figure 23).

As figure 23 shows, the expert obtained policy is not symmetric in both cases. It means that the expert performs better for negative initial angels. This is natural since we can see people who have more concentration on one of their body sides. The performance of the simultaneous learning can be seen more clearly on the trajectory paths.

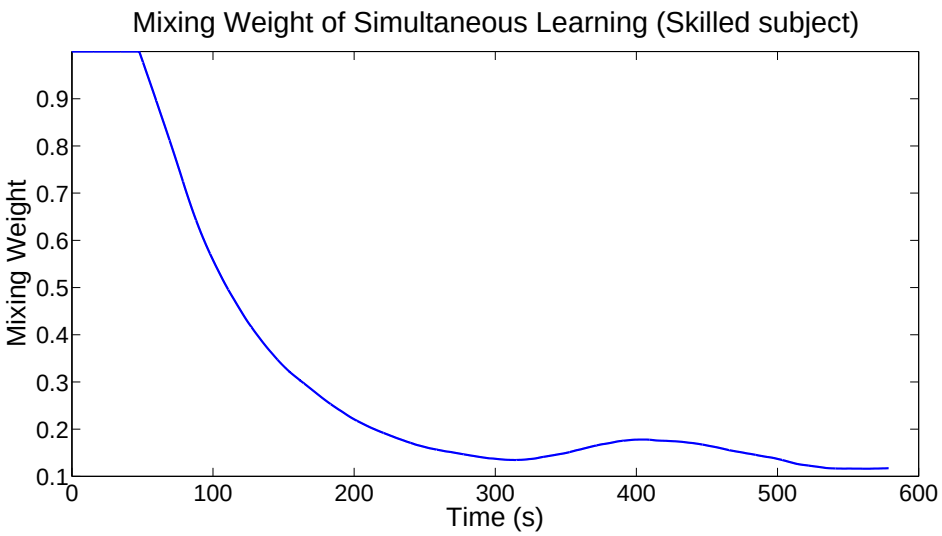


**Figure 17:** The mixing weight dynamics.

Figure 24 and 25 demonstrate the trajectory of the expert in sequential and simultaneous experiments, respectively. Although they seems similar in the spanning the state space, however, there is a huge difference in the autonomous trajectories. Figure 27 and 26 show the autonomous policy trajectories which is trained by sequential and simultaneous experiments. The figure 26 clear show that the autonomous policy trained by simultaneous could manage to not fall from the negative angles. And, also it is more concentrated on the center of state space ($\theta = \dot{\theta} = 0$) which the most desired points. On the other hand, the sequential autonomous policy, similar to the training part (see 27) has poor performance. The zero control trajectory also illustrated in figure 28 which is symmetric and directly quitting the acceptable state space range with almost same trajectory.

In simultaneous learning, the final action (sent to the robot) eventually change its source. At the beginning, it is most similar to the human action (mixing weight is 1)(see figure 29) and it is most similar to the machine learning's action at the end of the experiment (see figure 30).

It should be noted that during all experiments RFWR learning system were limited

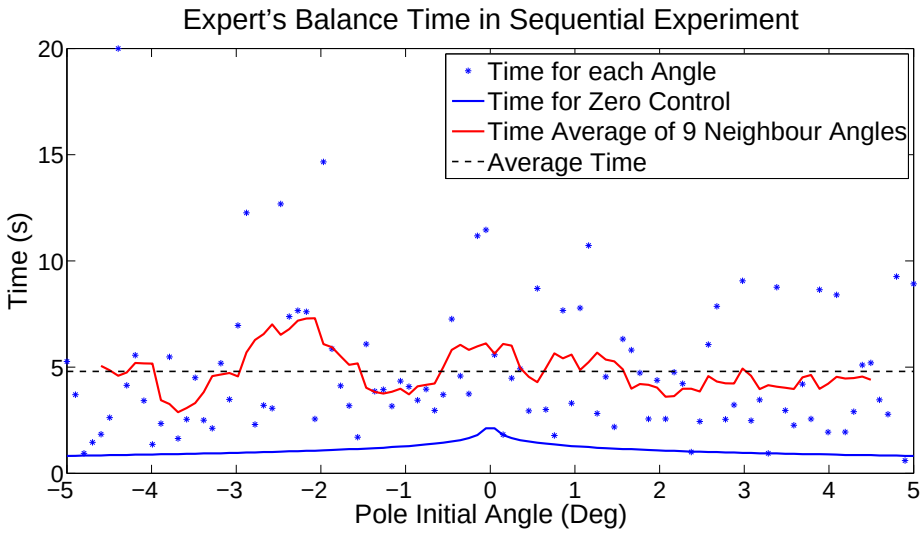


**Figure 18:** Performance of the expert while performing the experiment in sequential experiment.

to 100 units. Since the users were not consistent and the amount of data was not huge, this limitations help to have better generalization. This limitation equally applied for both sequential and simultaneous learning. The number of training point is 28938 for simultaneous and 23974 for the sequential one. The reason of difference is that in simultaneous learning performance of the expert was better and therefore the expert lasted longer over 100 trials.

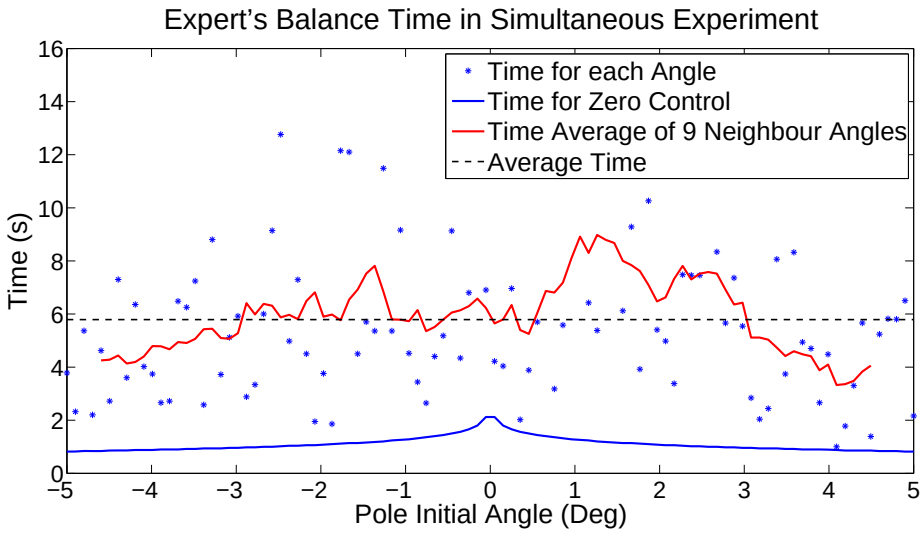


**Figure 19:** Performance of the expert while performing the experiment in simultaneous experiment.

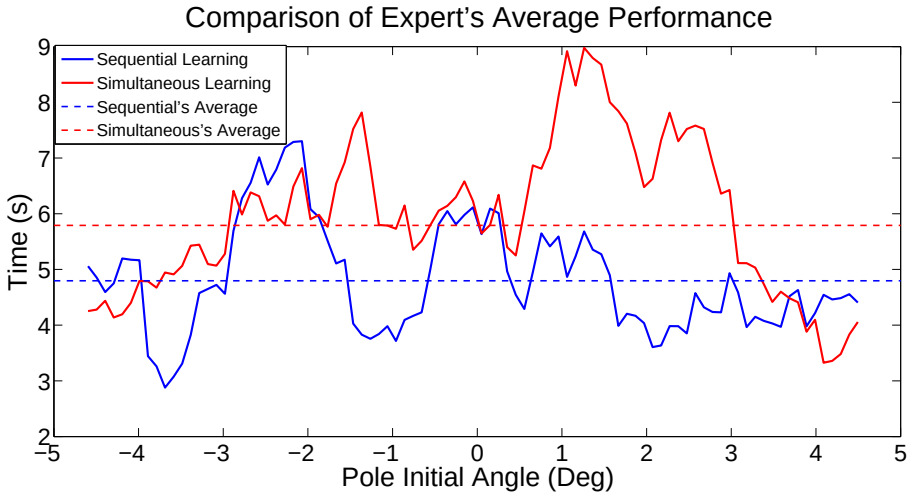


**Figure 20:** Comparison of the performance of the expert while performing the experiment (simultaneous vs sequential).

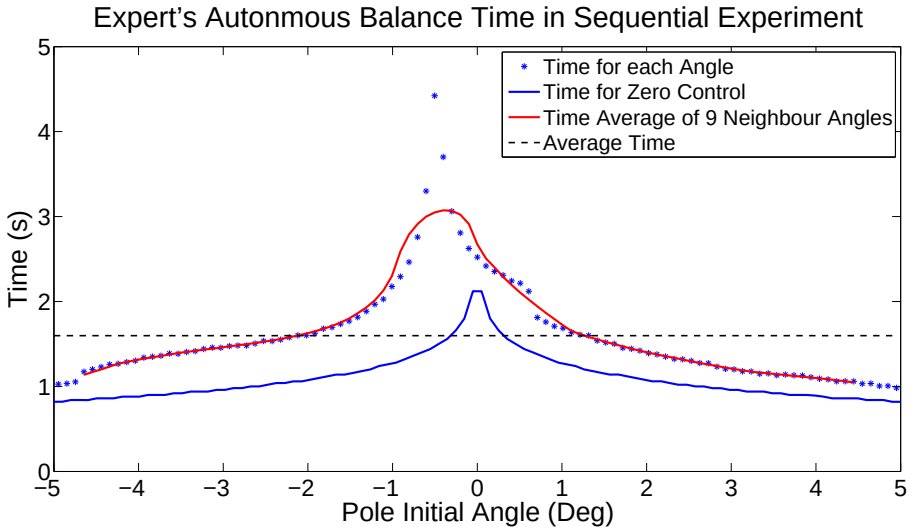


**Figure 21:** Performance of the Autonomous Controller trained by expert in sequential experiment. The Performance is shown for each individual point.

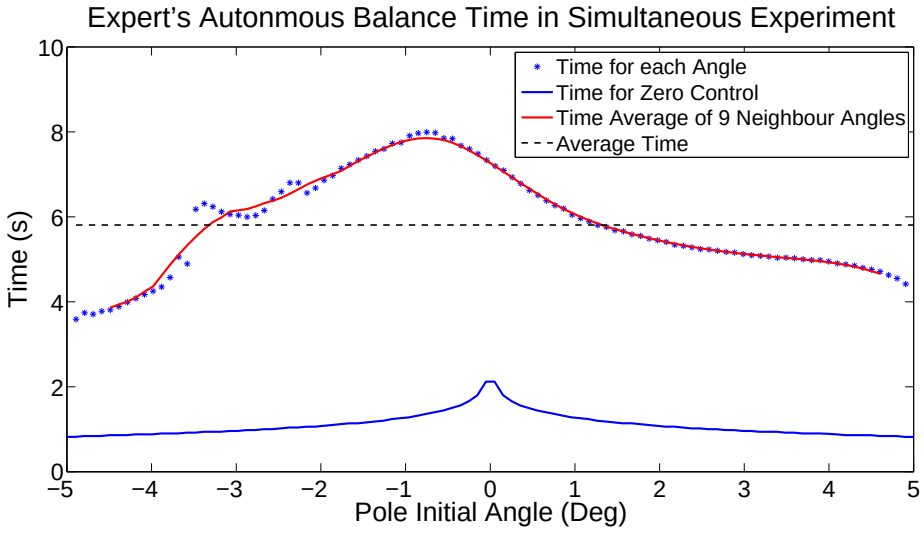


**Figure 22:** Performance of the Autonomous Controller trained by expert in simultaneous experiment. The Performance is shown for each individual point.

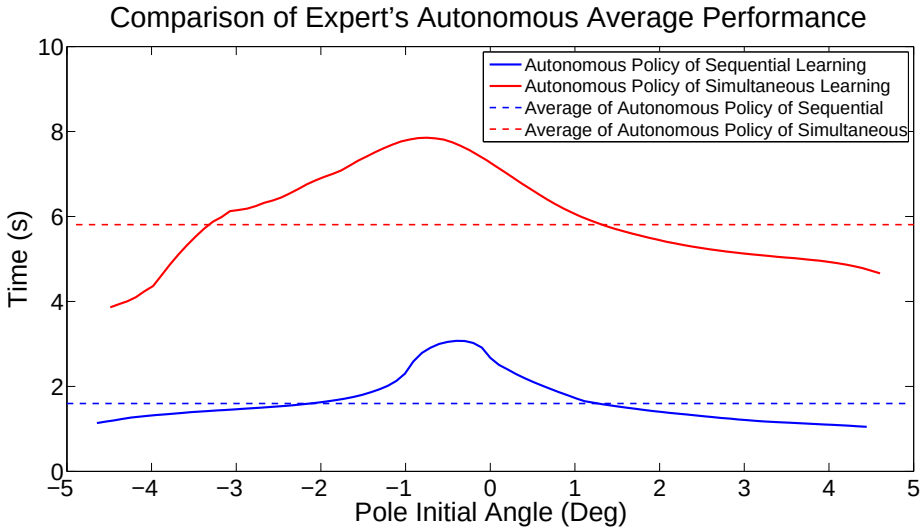


**Figure 23:** Comparison of Simultaneous and Sequential Autonomous Controller's performance trained by expert. The Performance is shown for average around each point and total average.

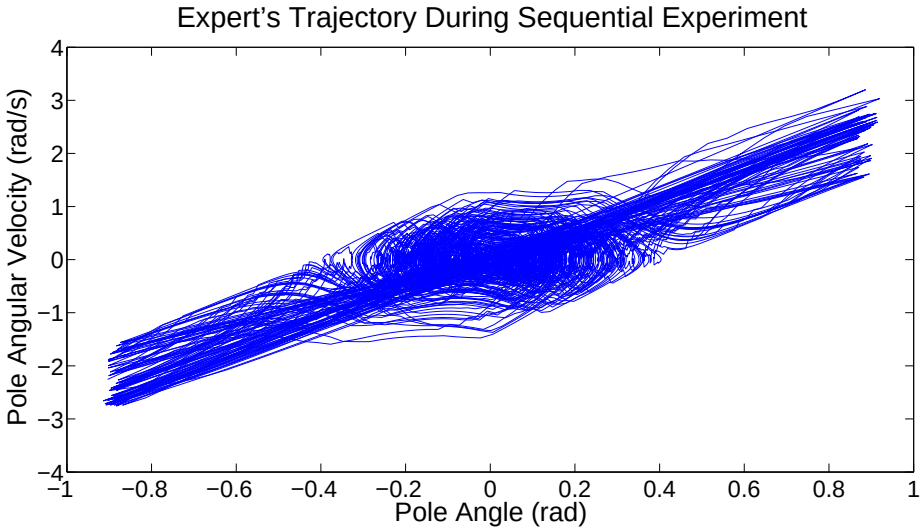


**Figure 24:** Trajectory of pole angle and pole angular velocity during the sequential experiment (Expert subject).

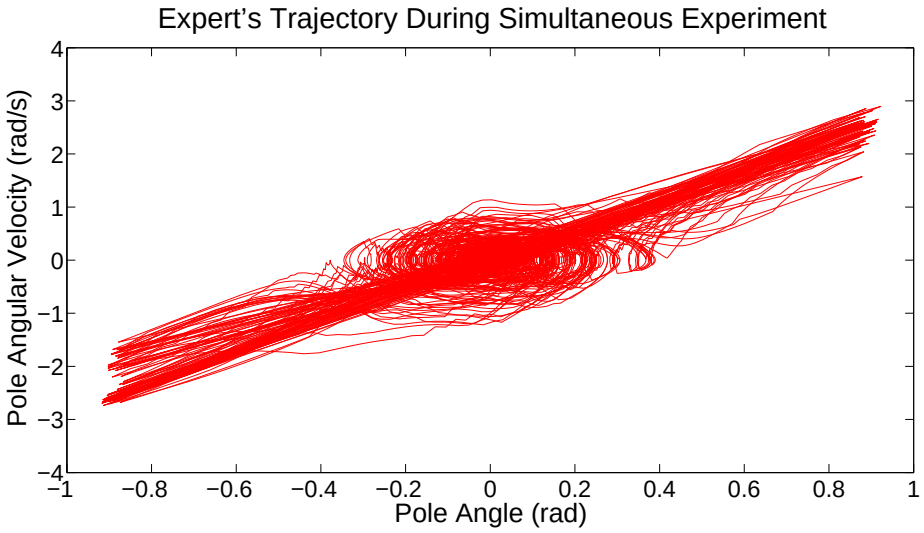


**Figure 25:** Trajectory of pole angle and pole angular velocity during the simultaneous experiment (Expert subject).

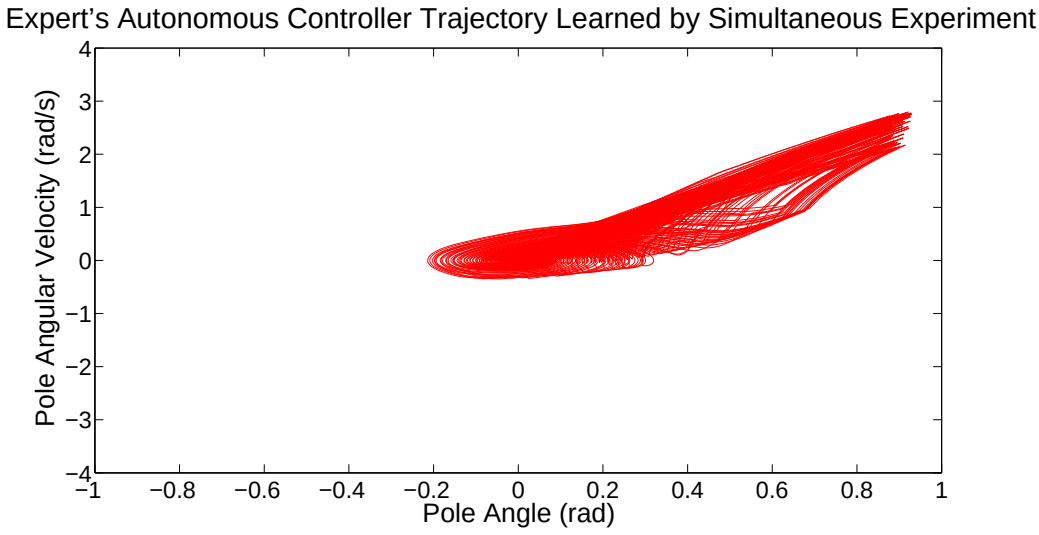


**Figure 26:** Trajectory of pole angle and pole angular velocity of autonomous controller learned by sequential method (Expert subject).

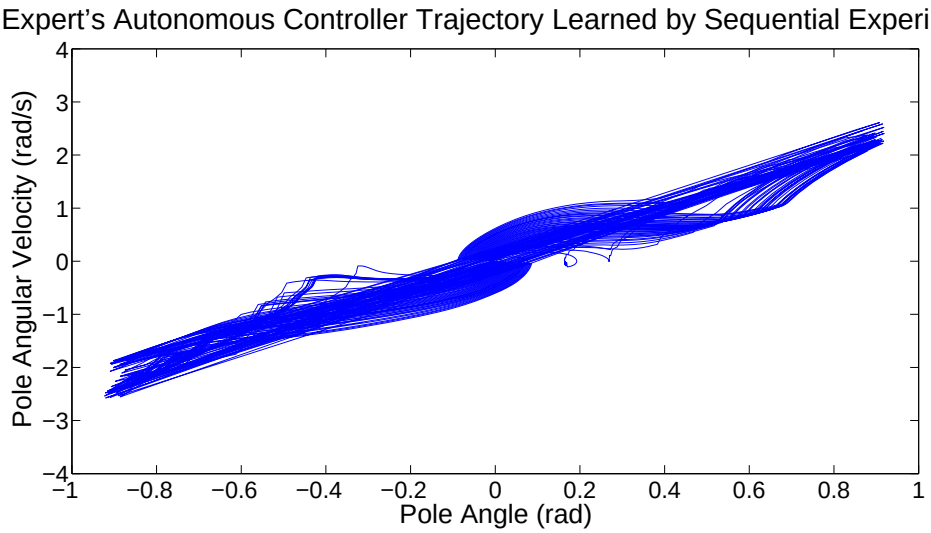


**Figure 27:** Trajectory of pole angle and pole angular velocity of autonomous controller learned by simultaneous method (Expert subject).

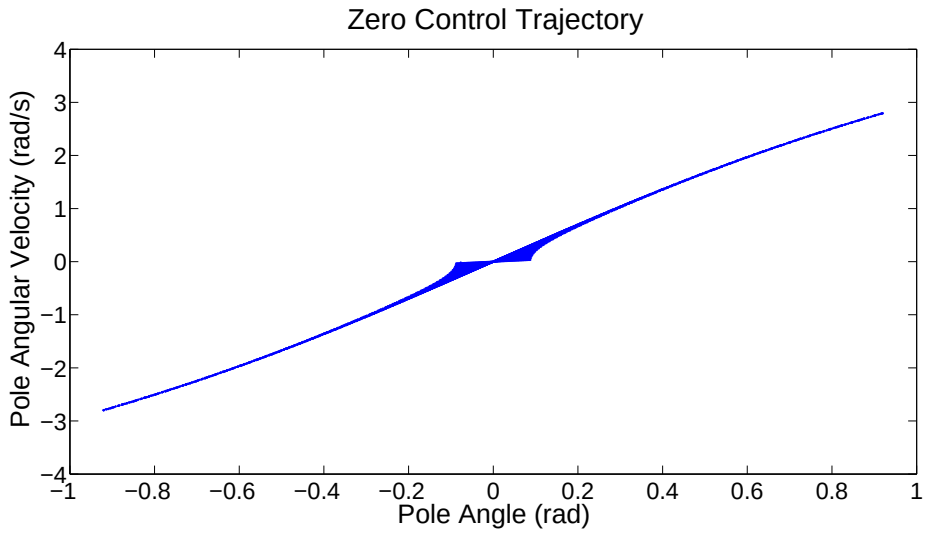


**Figure 28:** Zero Control trajectory.

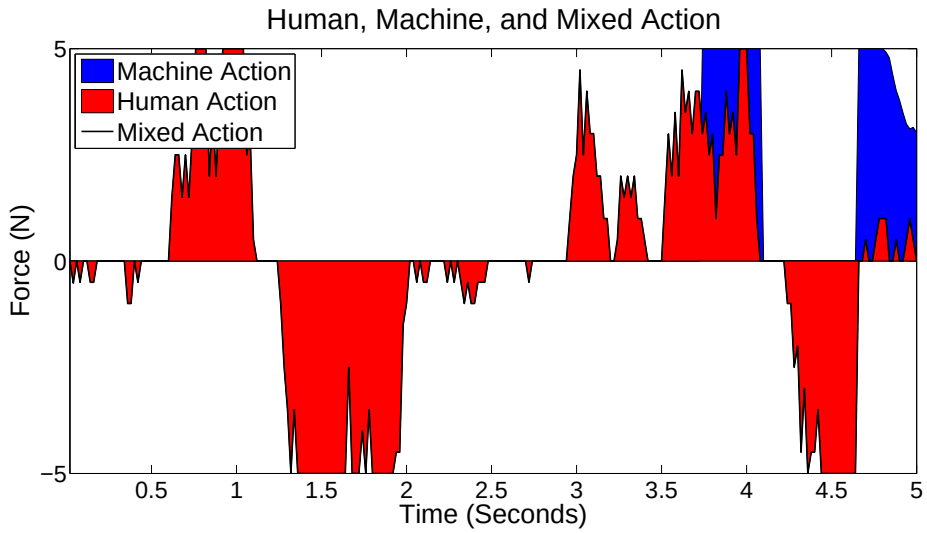


**Figure 29:** Human, machine and the mixing (net) action. As can be seen the mixed action is exactly following the human action.

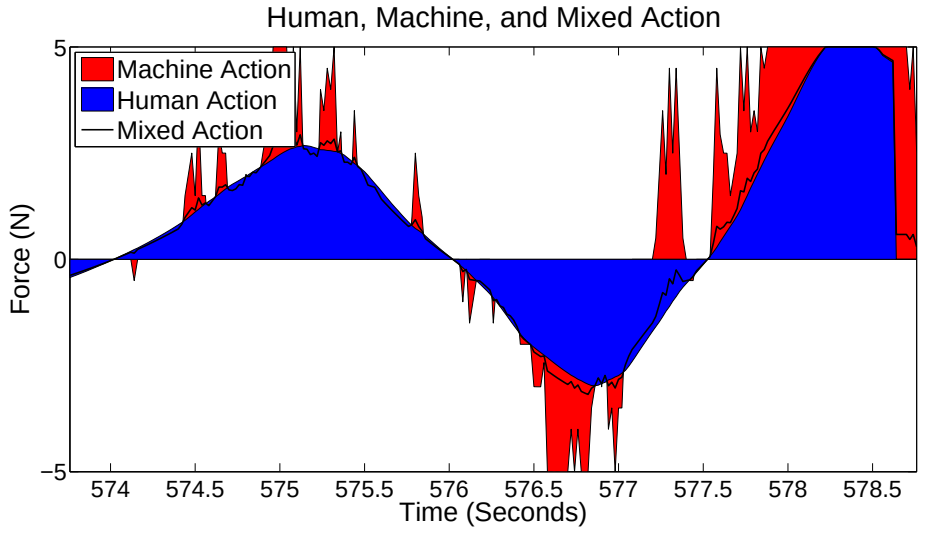


**Figure 30:** Human, machine and the mixing (net) action. As can be seen, at the end of the experiment, the mixed action is almost follows the machine learning's action.

### 3.2.5 Subjects Results

The experiments are repeated with two groups (simultaneous and sequential) of naive subjects to evaluate the framework. Each group consists of 4 people which randomly selected. Every subject had 20 trials as introduction to the balance task. And, after that, they had 100 trials (randomly released from [-5,5] initial angles) as the main experiment. The results are presented as their autonomous policy's performance. The simultaneous group subjects are told that the computer will help them to avoid any confusion of changing in dynamics of the task.

#### *3.2.5.1 During Training*

The balance time during training process is illustrated in figure 31 and 32. The results show that the subjects performance were not so different at the beginning of the experiments. We can even say that the simultaneous group performance on average was worse than the sequential one. Since, the subjects were naive they could not reach very high success rates. The figure 33 shows their success rates. As can be seen, different subjects have different skill level. Therefore, the subjects cannot reach to same level of mixing weight at end of 100 trials (see figure 34)

#### *3.2.5.2 Autonomous Controller*

During the learning, the subjects'policy obtained with RFWR method. After the learning, autonomous policy of the subjects are performed over 100 random initial points between [-5, 5] degree. The results show that the autonomous controller of the simultaneous group keeps the pole balanced (average 2.50 seconds and standard deviation 1.26) longer than the sequential one (average 1.37 seconds and standard deviation 0.39) (figure 35). Figure 36 shows individual performance results. The subjects are sorted based on their autonomous controller average balance time. The graph shows that each subject from the sequential group got a better record compared to the sequential one. Since, these are the final autonomous performance results. The

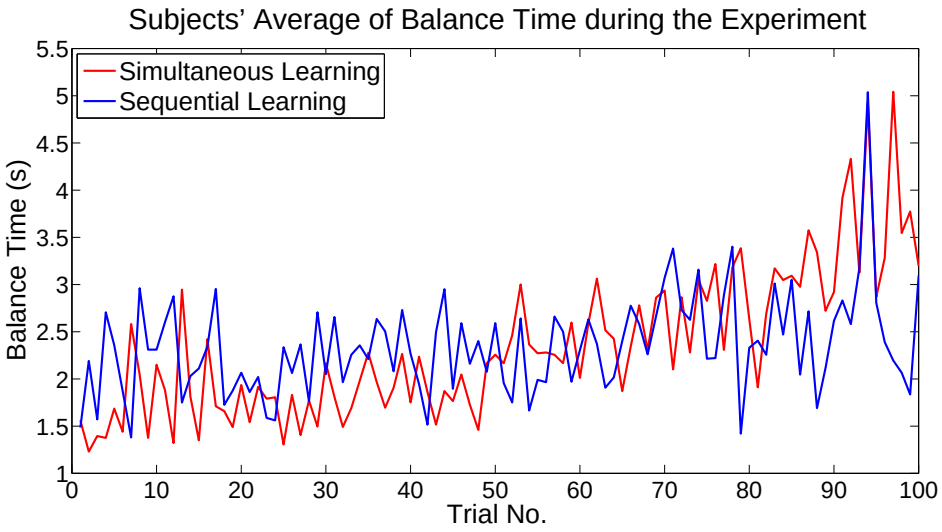


**Figure 31:** Average balance time of subjects in each group. Both groups performance increases by time.

autonomous performance of the subjects are evaluated in series of 10%, 20%, of total time to evaluate their progress. Figure 37 shows the autonomous performance of each subject before full training. Except one outlier subject all subjects are constantly improved their performance. The sequential group's performance also increases; however, it is not as significant as the simultaneous group (see figure 38).

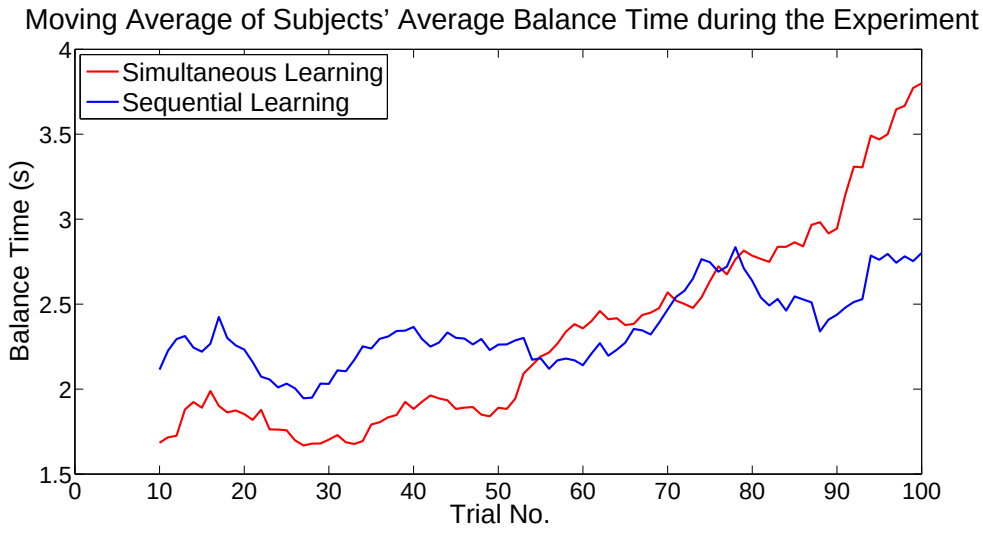


**Figure 32:** Moving average (window size=10) of average balance time of subjects in each group. The simultaneous group subject started with lower performance and ended up with better performance compared to the sequential group.

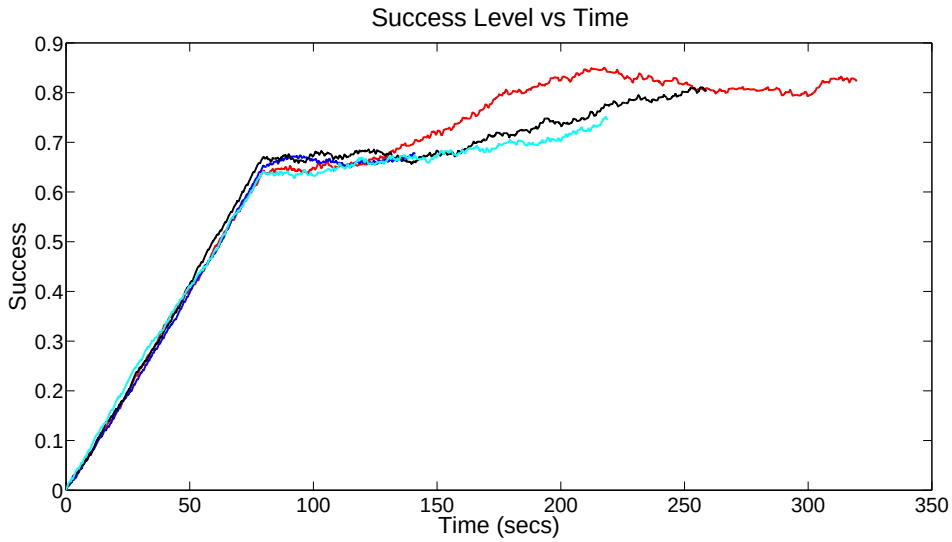


**Figure 33:** Success measurement of simultaneous group. Each color shows a different subject. Their skill and learning speed is different which lead to finish the experiment with different success level.

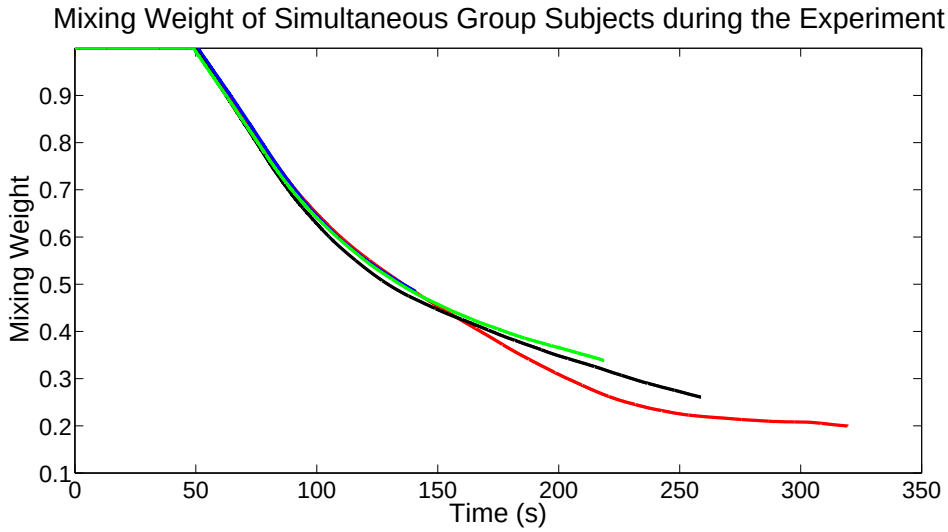


**Figure 34:** Mixing weight for different subjects. Each color shows a different. subject. The subjects' performance The curves finish in different time

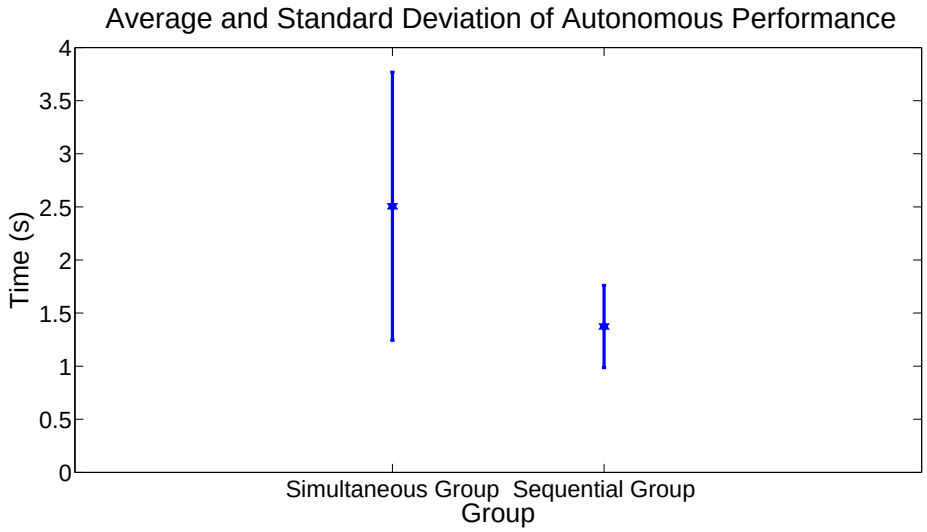


**Figure 35:** The average and standard deviation of each group's autonomous performance

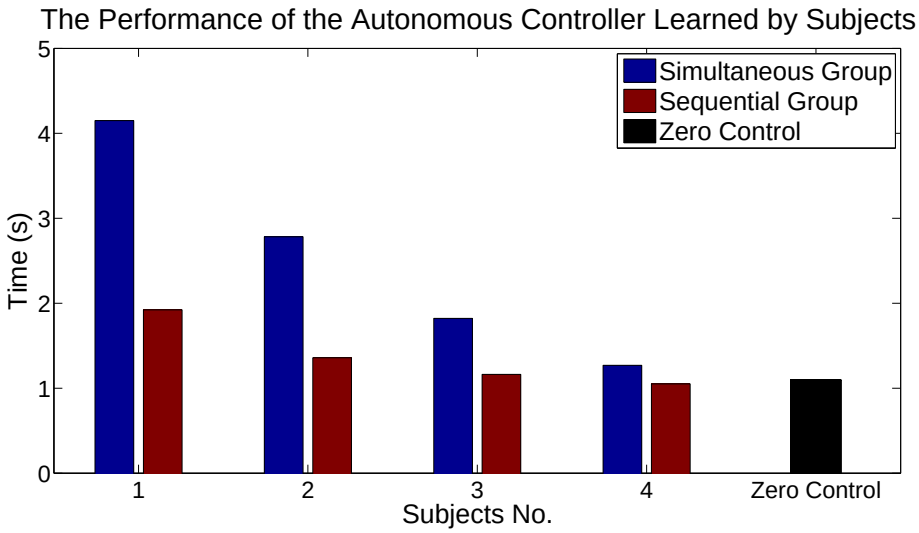


**Figure 36:** Comparison of the autonomous performance of the subjects. The subjects are sorted based on their autonomous policy time that can hold the pole.

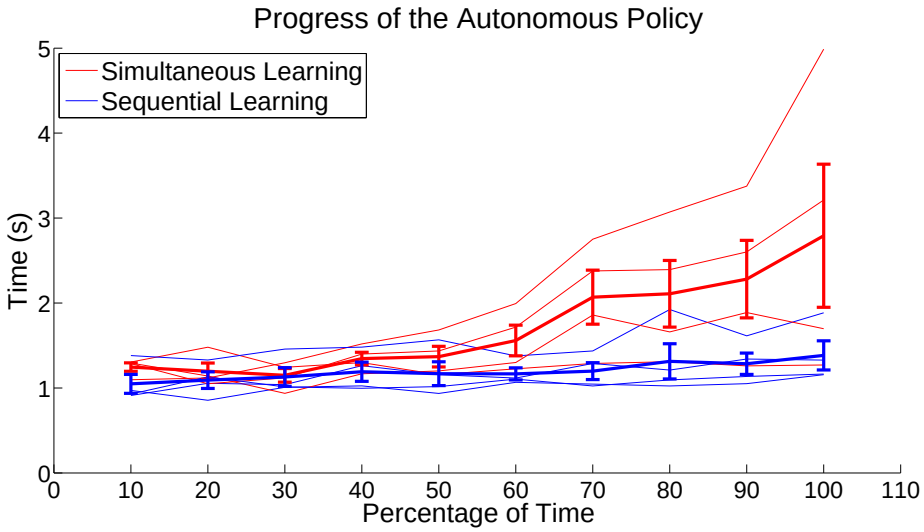


**Figure 37:** Average autonomous performance of the subjects before full training. The red line are related to simultaneous group and blue lines are for sequential group. Simultaneous group autonomous performance constantly increase by time. However, the sequential groups autonomous performance is improved little.

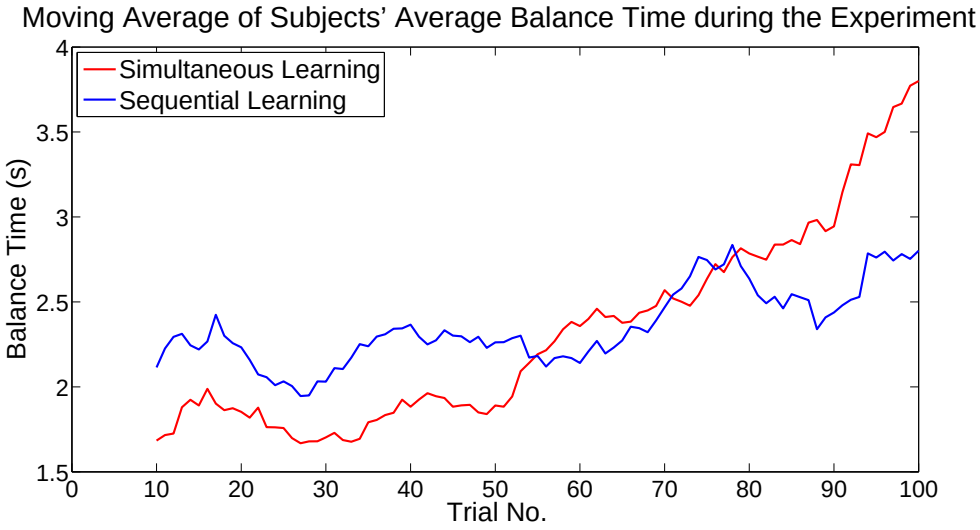


**Figure 38:** Average improvement of autonomous performance by time. The improvement is considered respect to the first 10% of the experiment. The simultaneous group had 155% improvement, and the sequential group had only 33% improvement

## *3.3 Speed-Up Task*

The speed up task is a new task which is introduced by this thesis author. In the *ball and beam* setup (see figure 39), a metal ball can freely roll on a track (i.e. beam). One side of the beam is fixed and the other side is connected to a servo motor via a lever. By controlling the position of the servo -which leads to change in the angle of the beam- the ball can be made to move to a desired position on the beam. We designed a simple task (called *speed up* or *maximum kinetic energy*) to analyze the algorithms we develop for simultaneous learning. In this task, the goal of the demonstrator is to roll the ball back and forth with maximum possible speed without hitting the two ends of the beam. The demonstrator action (mouse movements) is converted to servo motor voltages via a fixed gain. Here is the simplified dynamics of the system [6]:

$$\frac{\theta(s)}{V_m(s)} = \frac{K}{s(\tau s + 1)} \tag{24}$$

$$\frac{X(s)}{\theta(s)} = \frac{K_{bb}}{s^2} \tag{25}$$

Where, $\theta(s)$ is the angular position of the servo motor, $V_m(s)$ is the voltage given by the controller, $K$and $\tau$ are nominal servo motor parameters.

The controller generally follows the equation 26.

$$V_m = g(x, \dot{x}, \theta, \dot{\theta}) \tag{26}$$

Where $g$ is the desired controller function. However, it is possible to simplify the controller function. Since the dynamic of servo motor is fast enough $\tau = 0.0248 < T$ (sampling time) and the beam angle change between -5 and 5.

To define the success function first we need to define instant reward as equation 27.

[6] http://www.quanser.com/Products/Docs/1403/Ball-and-Beam-Courseware-Sample-for-LabVIEW-Users.pdf

$$R(t) = |\dot{x}(t)| - step(x_{\min} - x(t)) - step(x(t) - x_{\max}) \tag{27}$$

Where, $R$ is the reward, $x_{\min}$ and $x_{\max}$ are the limit which we are allowed to roll (i.e. outside of this limit is considered as hitting of beam edges). Then,

$$success = h(\frac{\overline{R^t_{t-\alpha}} - R_{\min}}{R_{threshold} - R_{\min}}) \tag{28}$$

**Figure 39:** Quanser Ball and Beam setup. Image from (http://www.quanser.com/).

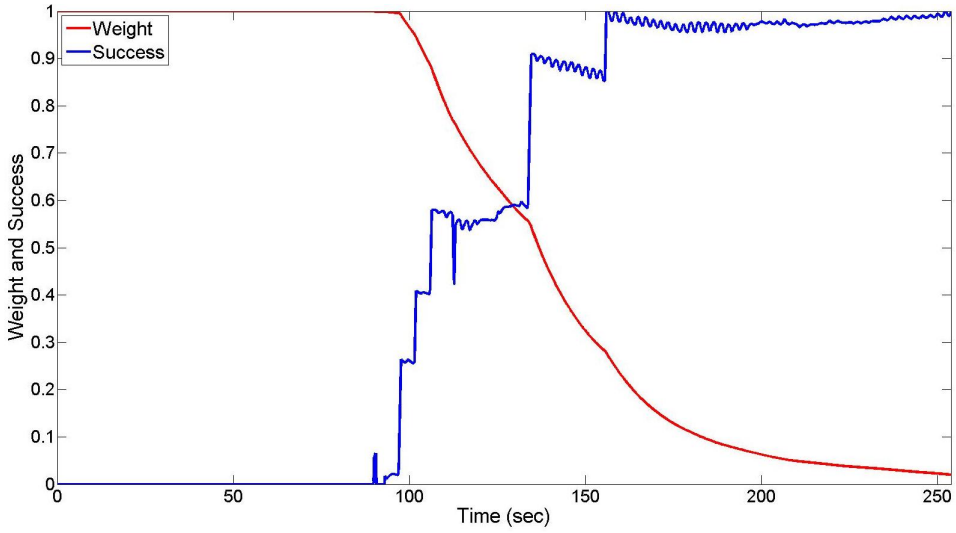


**Figure 40:** Success rate and weight change during the expert learning.

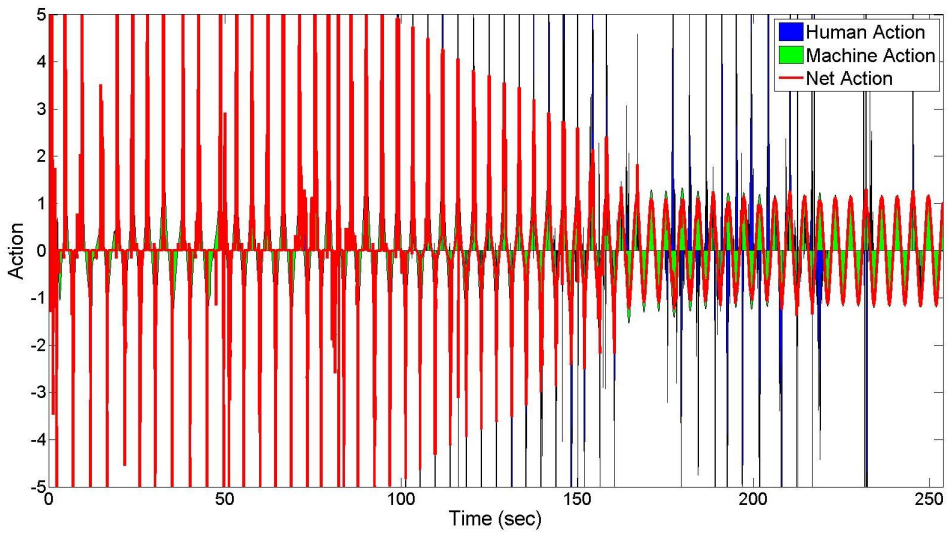


**Figure 41:** Human, Machine, and net (mixing action) of expert user in speed-up task.

The implemented results indicate that we can transfer human skill to the machine in very short time. In less than 5 minutes an autonomous controller could be obtained which can perform the task quite well without human guidance. Figure 40 shows the mixing weight and success level. Human, machine learning and mixed action can be seen in figure 41 that is generated during simultaneous learning experiment. Note that the mixing weight decreased from 1 (fully manual) to 0 (fully automatic) based on the local success measure that increased and converged to one during the session. The learning process and the autonomous control performance can be seen in this video clip[7].

[7]https://youtu.be/BtewbdaRQc0

# CHAPTER IV

# SIMULATION OF REINFORCEMENT LEARNING AND SUPERVISED LEARNING CONTROL MIXING

## *4.1 Introduction*

In previous chapter, a framework is proposed which can transfer skill from human demonstrator to robot. The expert subject's performance shows that using the proposed framework human can complete the task. Moreover, to understand whether it is advantageous over sequential learning, the framework was examined by two groups of subjects. However, to analyze the proposed framework more closely, human demonstrator is substituted by reinforcement learning. The intuition behind this choice is similarity of RL method and human learning.

In this chapter, similar to the previous chapter, two simultaneous and sequential frameworks are compared to each other. In sequential framework, the RL agent obtains a policy and a supervised learning method learns the policy without contribution in control (i.e. no mixing control). There is one considerable difference between this chapter and previous one. When a human demonstrator does the task in the sequential experiment, the supervised learning was capturing the human's policy. In other words, to reproduce the human policy it was necessary to use a supervised learning method. In the RL agent, there is two options to obtain the policy: 1) the supervised learning (similar to human sequential learning) 2) the RL policy function which is more desirable since the RL agent records its own policy and at the end of learning the policy is available. And, it is used as the final autonomous policy of sequential learning (figure 42). On the other side, there is a simultaneous learning framework which contains of an RL agent and a supervised learning. The supervised

learning not only learns in the background, but also eventually contributes in the control signal and gets more shares when the task progress according to the success criteria (figure 43).

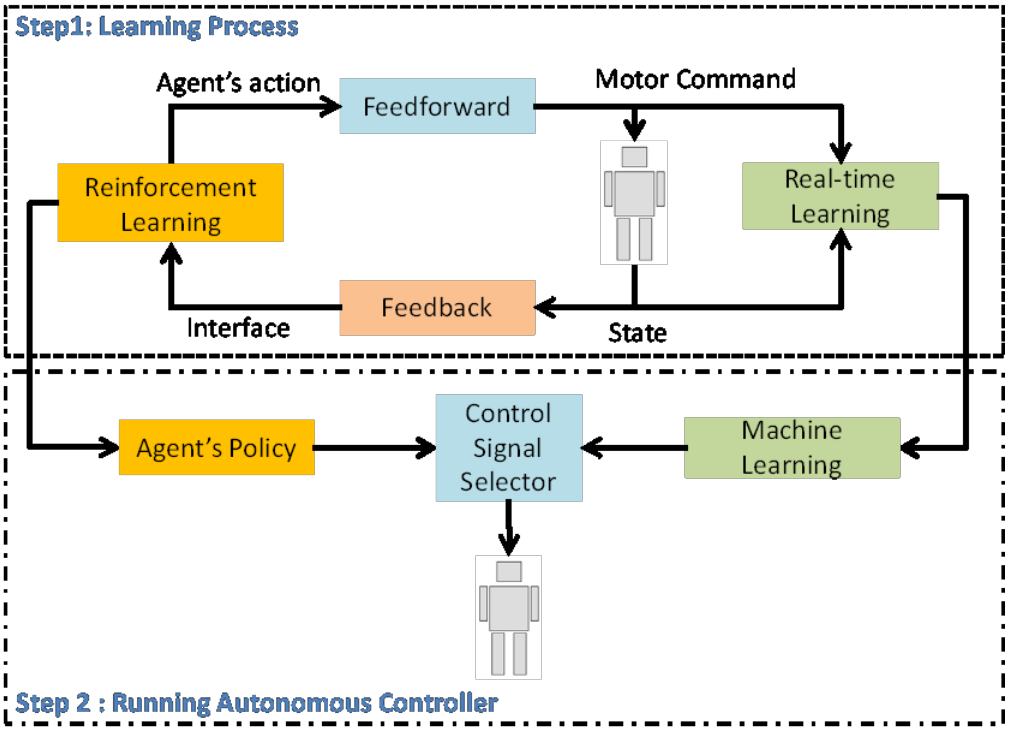


**Figure 42:** Sequential Reinforcement-Machine Learning framework is illustrated. The RL as model of human demonstrator controls the robot in real-time to achieve a desired goal through the robot . Simultaneously, the ML learns to imitate the RL policy. The net control the robot receives is obtained by the RL. After the learning is finished (step 1), there are two options for the autonomous controller. a) RL agent's policy and b) ML policy. There is no mixing in the sequential framework and the controller is obtained after learning finished.

## *4.2 Reinforcement Learning-Machine Learning Simultaneous Learning System*

A framework is proposed that aims to engage reinforcement learning and machine learning simultaneously to improve the effectiveness of reinforcement learning (as a model of human learner) skill transfer to machine learning (which is te learning core of the robot). This may be seen again analogous to the learning process between a

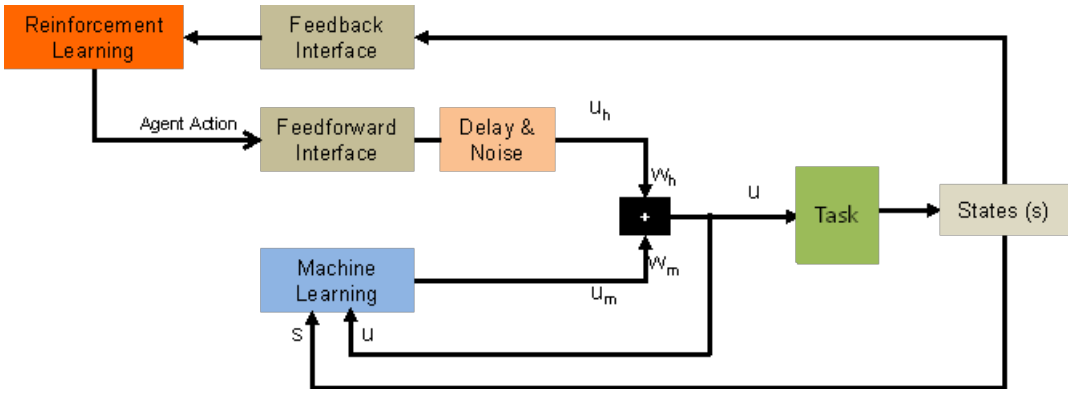


**Figure 43:** Simultaneous Reinforcement-Machine Learning Framework is illustrated. The RL as human demonstrator controls the robot in real-time to achieve a desired goal through the robot . Simultaneously, the ML learns to imitate the RL policy. The net control the robot receives is obtained by the weighted summation of the RL and ML generated controls. The weighting is dynamically adjusted so as to pass the control to the robot when successful learning and task completion is achieved. At this point the task skill has been transferred to the ML and it can complete the task autonomously without requiring the RL.

human teacher and a human learner. The main purpose of this chapter is to support the human-robot simultaneous framework. The general concept is quite similar to the previous chapter. The supervised learning method (learner) tries to mimic the teacher's action gradually by getting help from the agent (teacher) when needed. As the learner becomes better at satisfying the tasks objectives, the teacher involves less. However, when this deteriorates performance the teacher intervenes back and issues corrective actions. However, the reinforcement learning agent which starts without any model of the task first learns itself and gradually transfers the knowledge to the machine learning system. In this framework, likewise, the reinforcement learning has the full control of the task in the very beginning, and gradually the control is shifted to the machine learning based on the overall performance, so if the performance degrades the reinforcement agent's share in control increases. When the success objectives are completely satisfied, the machine learning becomes the only controller. At this point, the learned policy can be preserved as it is and deployed later for full

autonomous execution of the task without reinforcement learning. Figure 43 presents our Reinforcement Learning-Machine Learning simultaneous learning framework. The RL agent controls the robot in real-time via fixed feed-forward interface, and observes the robot state through a feedback interface. At the same time, the machine learning system starts to mimic the RL's policy and injects its own control gated through a weight parameter. The final action (u) is simply the linear combination of the RL and machine learning action ($u_h$,$u_m$ respectively):

$$u = wu_{RL} + (1 - w)u_{ML} \tag{29}$$

Where $0 \leq w \leq 1$ indicates weight or share of the human action on the overall control. The key design issue is to define a $w$ dynamics to facilitate effective skill transfer avoiding oscillations between RL and ML control. For this we propose to use a time-windowed success indicator to determine the dynamics of $w$. The proposed framework does not specify the details of the involved feedback or feed-forward interfaces and current implementation which studies simulation of the task assumes the feedback and feedforward systems work properly.

### 4.2.1 Mountain Car Task

Mountain car is an underactuted robotic task (figure 44) which aims to bring a car to the top point of a hill by moving it back and forth to accumulate sufficient energy [12]. The task has to be completed in less than or equal to 500 time steps. Otherwise, the car resets to the initial point. The actions are selected among [-1 0 1] which mean backward, no, and forward throttle. The discrete equation of the mountain car are:

$$v_n = v_{n-1} + 0.001u - 0.0025\cos(3x_{n-1}) \tag{30}$$

$$x_n = x_{n-1} + v_n \tag{31}$$

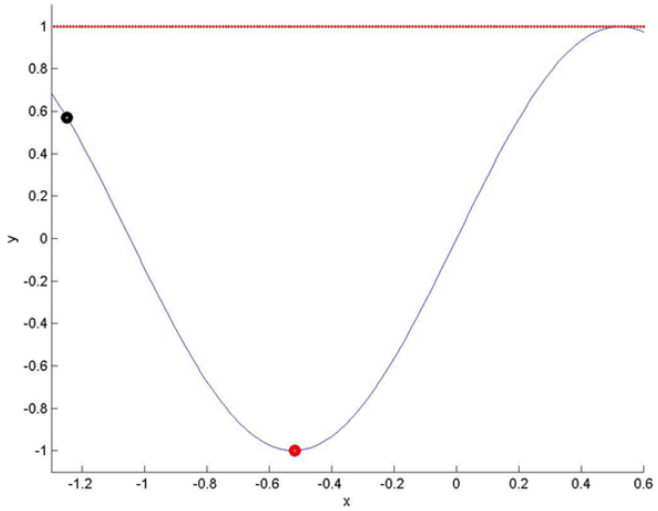


**Figure 44:** Mountain Car Problem. The goal is to reach to the top right side.

Where, $v_n$ and $x_n$ are the velocity and position of the car in time step n , respectively. The range of the position is between [-1.2 , 0.5], and the range of velocity is between [-0.07, 0.07]. Position $x = -1.2$ is an inelastic wall. That means when car reaches to the position, the speed set to zero. The height of the car is defined by the equation 35 as defined in original problem [12].

$$h = \sin(3x) \tag{32}$$

which describes the curve that the car moving on it. So, the desired state manifold is given by $x \geq 0.5$.

The initial state of car is $[x_0, v_0] = [-0.7, 0]$. The reason that we did not select $[x_0, v_0] = [-0.52, 0]$ (deepest point) as the initial point is that there has to be more discrete states over there to avoid discretization error. For example, reference [33] applied 1751x151 grids to find an accurate estimation of the optimal value function and they solve the infinite horizon MDP problem with Value-Iteration methods. In this thesis, however, we used 100x100 grids to discretize mountain-car state spaces

(for both simultaneous and sequential learning). This much discretization is good-enough that value-iteration converged results can complete the task and also does not have heavy computational load.

The control policy is captured by a time independent function of the mountain-car state $u = g(x, v)$. The goal of the RL-ML learning is to have the RL produce control data points than can be learned by a machine learning algorithm so that g is obtained and can be used as an autonomous controller for the task. Here $u$ is the controller output.

### 4.2.2 Reinforcement Learning

Q-learning [13] as one of the well-known reinforcement learning is applied here to model human learner. Reinforcement Learning is commonly known as a method to model human behavior [34, 35]. In this thesis, it is not claimed that the reinforcement learning which is used is a perfect model for a human. The main goal is to support the main framework of simultaneous learning and give an intuition of how learning process shapes and why it can be beneficial.

Q-learning agent updates the state-action value function by an explorative-exploitative agent (e-greedy usually) [12]. The update happens with a learning rate and a forgetting factor as following formula [12]:

$$Q^{'}(s,a) = Q(s,a) + \alpha(R(s,a) + \gamma \max_{a'} Q(s^{'}, a^{'}) - Q(s,a)) \tag{33}$$

Where $Q$ is the state-action value function, $s$ and $a$ are the state and action, respectively. $\alpha$ and $\gamma$ are learning rate and forgetting factor, respectively. $R$ is the reward function and $'$ symbol represent the next time. The optimal policy of Q-learning is [12]

$$\pi(s) = arg\max_{a} Q(s,a) \tag{34}$$

Where $\pi$ is the policy, and * refers to optimal solution.

### 4.2.3 Machine Learning

An important notion of human-in-the-loop skill synthesis is that 'data'is the key not the underlying machine learning algorithm, as the framework allows generation of large data sets as the human is placed in the control loop. So, we do not focus on the specifics of the machine learning algorithm to represent g(), but instead underline the requirements for a generic machine learning algorithm for simultaneous learning: (1) support for online incremental learning, (2) robustness against over-fitting, and (3) low computational load. A good match for these requirements is the Locally Weighted Projection Regression [36].

### 4.2.4 Weight Dynamics

Weight dynamics concept equation is identical to previous part (equation 20). $\tau$ is also defined based equations 23. $f(success)$ is also defined same as the equation 7.

### 4.2.5 Success Measurement

In the mountain car system an intuitive success measure is the (average) original reward (equation 35). The goal in our mountain-car is to bring the car from near deepest point to the top of the hill, where the height achieves its maximum. So, representing the reward obtained as $r$, which is -1 for each step and 100 if it reaches to the top. However, the earlier framework [37] support division of the task into multiple state regions, the nature of mountain car does not need to apply state regions.

$$success_i(t) = l(\frac{\overline{r(t)}^t_{t-\alpha} - r^{\min}}{r^{threshold} - r^{\min}}) \tag{35}$$

where $h$ is given by

$$l(x) = \begin{cases} 0, & \text{if } x < 0 \\ x, & \text{if } 0 \leq x \leq 1 \\ 0, & \text{if } x > 1 \end{cases} \tag{36}$$

and $r^{\min}$ indicates the minimum average reward of the car (here is -1 means the car just oscillates without reaching to the top), $r^{threshold}$ is a constant threshold (acceptable average reward which is obtained here by try and error =0.05[1], and finally $\overline{r(t)}^{t}_{t-\alpha}$ is the moving average of the height within the past $\alpha$ time unit (here, 10000).

It should be noted that since the mountain car is episodic task, it is necessary to wait until the episode finishes to be able to judge about the success. Therefore, here time refers to episodes. Moreover, the success is defined to state-region free, then the weight mixing also becomes state-region free. Hence, the dynamical equations governing the weights is given by:

$$\tau \dot{w}(t) = -w(t) + 1 - l(\frac{\overline{r(t)}^{t}_{t-\alpha} - r^{\min}}{r^{threshold} - r^{\min}}), w_i(0) = 1 \tag{37}$$

### 4.2.6 Obtaining the Autonomous Controller

The task completion means w=0 and success=1 which refers that the skill is already transferred to the ML, when the weight approximately[2] reach 0 as at this time the RL contribution to control would be minimal. At this point, the parameters of the machine learning system is retrieved and saved as the autonomous policy that can be deployed at a later time. Since there is no prior knowledge that when the skill completely transfers, exploration of the RL agent overwrite an improper policy if it continuous when the task complete (w $\approx$ 0 and success=1) and learning can be stopped.

[1]However, this value is close to the optimal average collected reward.

[2]To reach exactly zero it is necessary to wait very long which is not necessary. Therefore, around zero is also proper for the purpose

## *4.3 Results*

The proposed simultaneous RL-ML learning framework for modeling of robot skill transfer was evaluated on a simulated mountain car system. The simulation sessions started with the car positioned in the $[x_0, v_0] = [-0.7, 0]$ . The learning is tested in sequential mode where the sole control is given by the RL; And, in simultaneous mode, where the RL and ML control sharing was employed based on proposed learning system. The open parameters for the proposed framework is the size of the moving average window ($\alpha$) that is used in computing the task success (success parameter). It is selected large enough to capture the success measurement behavior. Moving average windows is selected as 10000 episodes for the experiments reported in this paper. Another parameter is the $h^{threshold}$ is selected 0.05 (out of [-1,1] interval). The other open parameters related to Q-learning are exploration rate (e-greedy), forgetting factor $\gamma$ and learning rate $\beta$which are selected as 10%, 0.99 and $\beta(episode) = \frac{10}{(episode+100)^{0.51}}$ . The learning rate is decreasing by episodes to be valid in the convergence of Q function condition which is [13]:

$$\sum_{episode=1}^{\infty} \beta(episode) = \infty \tag{38}$$

$$\sum_{episode=1}^{\infty} \beta^2(episode) < \infty \tag{39}$$

Since this research simulates the human-robot simultaneous learning, and human demonstrator already has much more better prior knowledge than reinforcement learning, a gate is set to start learning when the success level reaches to 0.5.

Average reward is used to evaluate the performance of the autonomous policies obtained based on the sequential and simultaneous learning. The obtained policies are evaluated for both sequential and simultaneous scenarios in the trained initial point $[x, v] = [-0.7, 0]$ and 240 different initial state as testing and theses points are selected

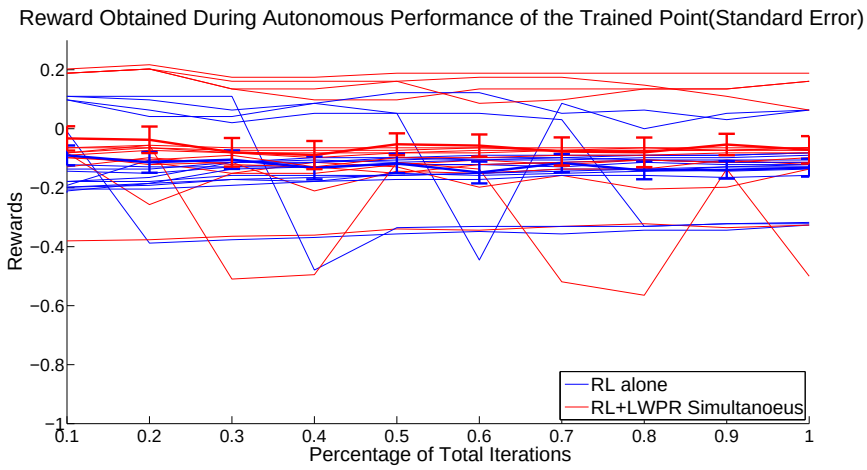


**Figure 45:** The figure compares Simultaneous Reinforcement-Machine Learning and the Reinforcement Learning alone (can be referred as sequential learning). The results presented here is the performance of the autonomous policy which is obtained from two frameworks. As can be seen in the figure, every 10 % of predefined learning time(iterations), the exploration is paused and the policy is tested and result is presented for one of each 10 %. The experiment is repeated 15 times for each simultaneous and sequential simulations. The results show that the average of simultaneous learning is higher.

uniformly over the state-space (15x16 grids). Moreover, to monitor the progress of the learning carefully, the performance of the obtained controller during the learning is also tested in 10%, 20%, ..., 100% of the total learning. Figure 45 show the performance of the autonomous controller for both simultaneous and sequential methods over every 10% multiple of predefined total learning time. The results which obtained over 15 runs for each group show that the average performance of the simultaneous method stays higher than the sequential one over the time.

Figure 46 compares generalization of simultaneous and sequential methods. For each initial point the reward is normalized respect to the maximum reward that is obtained for the point over 30 runs (both simultaneous and sequential). Then average of difference between simultaneous and sequential is presented a criterion to compare two methods. The value is always above zero which means that simultaneous method always does the task more efficiently.

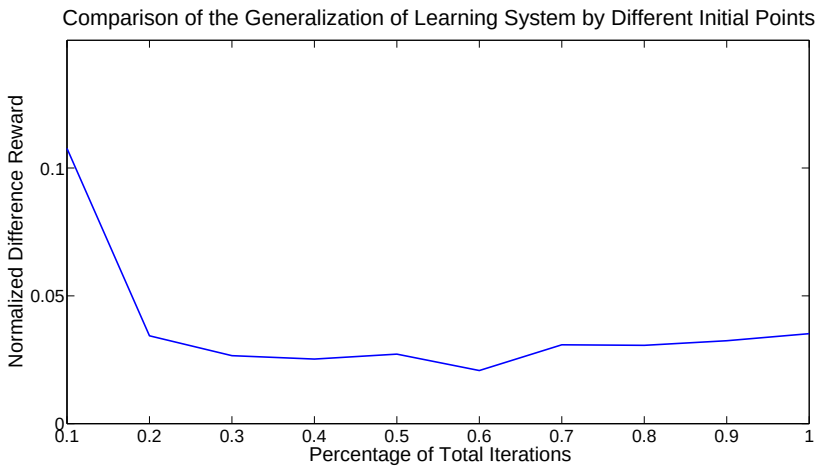


**Figure 46:** Generalization ability of Simultaneous Learning over Sequential Learning. This graphs shows the difference between normalized reward of simultaneous and sequential obtained by their autonomous controller during the each 10% of time passed from the total predefined time. simulations. Each point is the average of 240 different points in the problem stated obtained from 15 runs.

Figure 47 and 48 show the final value function from both methods. The value function of simultaneous learning seems to be slightly less explored. Since, by passing the time the machine learning algorithm takes the control and the RL lose the chance of exploration.

Figure 49 similar to the figure 46 show the generalization difference for each initial point using normalized reward difference. There are 240 colored squares which each square represent the normalized difference reward over 15 runs. The warmer colors (e.g. red) means autonomous controller of simultaneous method obtains higher reward and a cold color (e.g. blue) means the autonomous controller of sequential method obtains higher reward. As can be seen clearly there is the trajectory path mountain-car (seen in figure 47 and 48) is detectable in figure 49. It seems that when the test initial state are those points which are always included in the training path are done better by sequential method (It should be noted the [-0.7,0] is red means the simultaneous better as presented in figure 45). And, most of the other initial points which is not visited are done more efficient by the sequential method. This

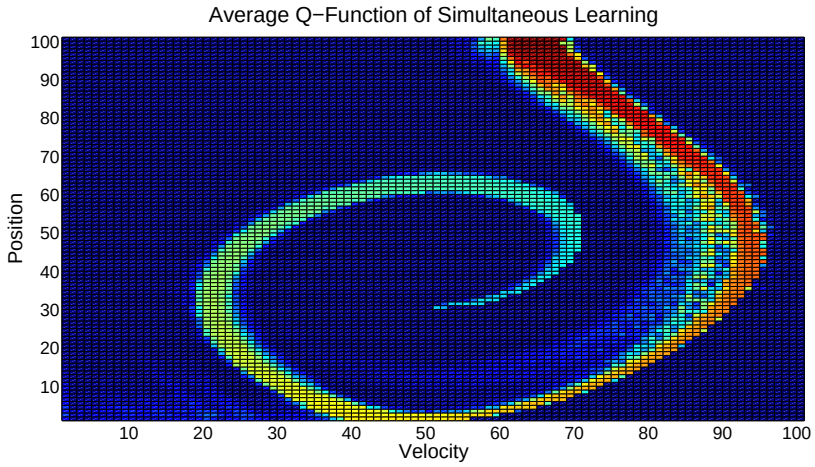


**Figure 47:** Q-function of the simultaneous learning. This is 100x100 discretization and the trajectory is the mostly explored path. The reason to have a sharp trajectory is that the learning is always start from the same initial point. As can be seen the simultaneous methods explore slightly less than sequential one.

simply explain that the explanation for this successful result is that RL gets help by generalization of machine learning.

Figure 50-53 presents the generated trajectories by autonomous controller from best run of simultaneous group and also the best run of sequential group. The simultaneous policy always generates wider spiral trajectories. That means the car reaches to the goal with faster speed. The success rate of the best simultaneous run is depicted in figure 54 . And, the mixing weight which is a function of the success is shown if figure 55.

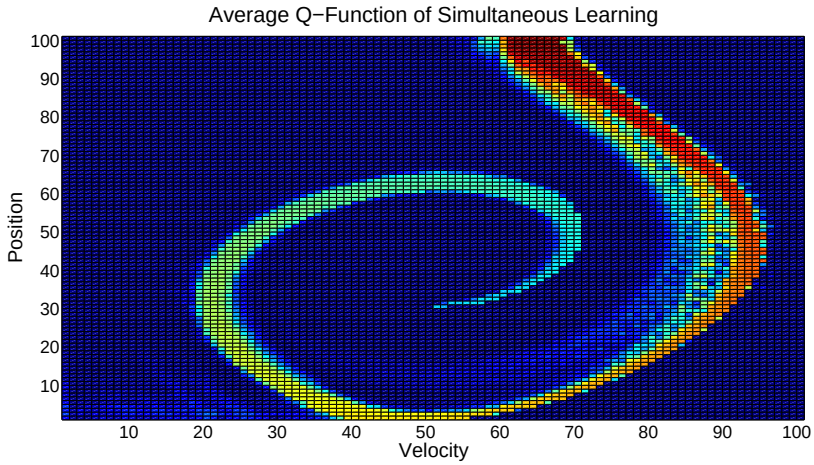


**Figure 48:** Q-function of sequential learning. This is 100x100 discretization and the trajectory is the mostly explored path. The reason to have a sharp trajectory is that the learning is always start from the same initial point. As can be seen the simultaneous methods explore slightly less than sequential one.

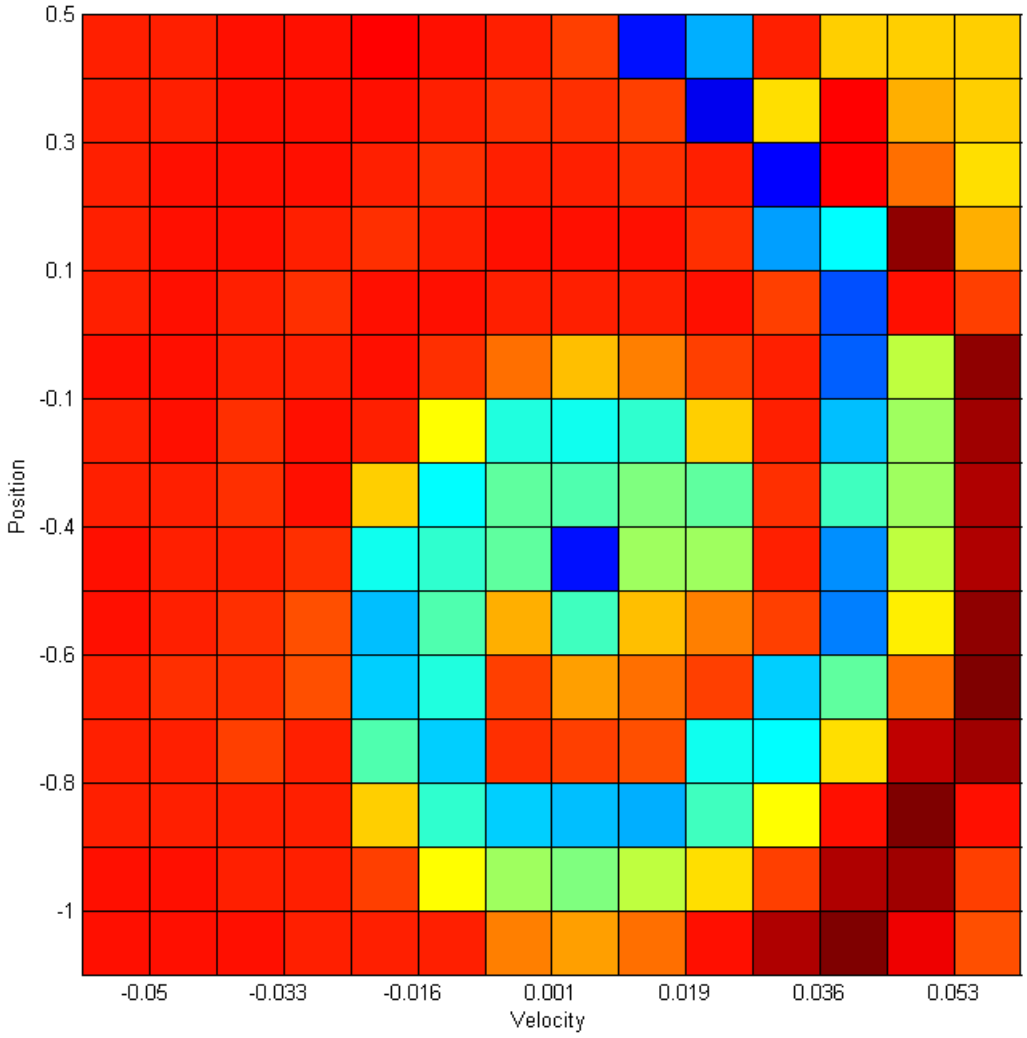


**Figure 49:** Normalized relative accumulated reward for comparing the simultaneous learning and and the sequential learning.. This figure similar to figure 46 shows the difference between the initial points over 15 runs for each method. As can be seen the trajectory of the mountain-car is visible with colder color.

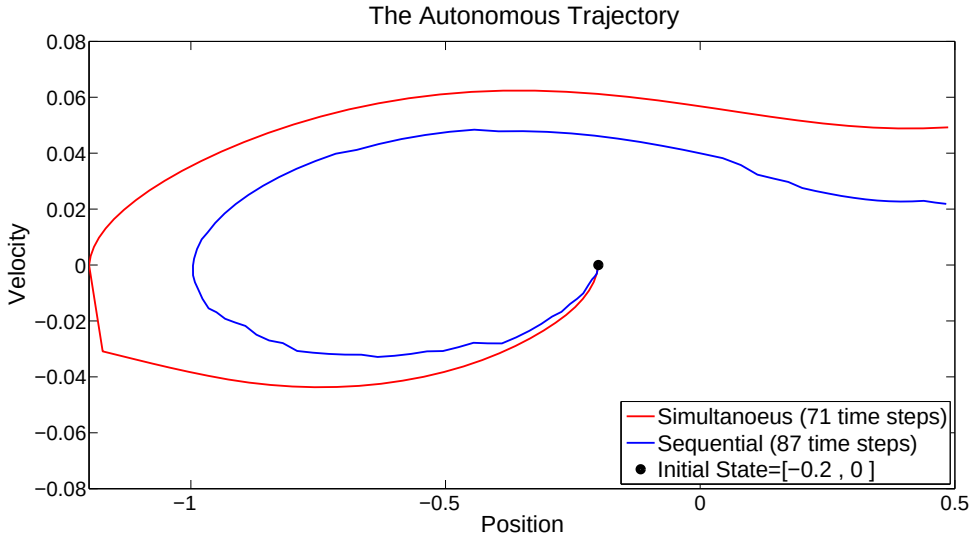


**Figure 50:** Comparison of the best Simultaneous run and the best Sequential run trajectories starting from the same point. This is testing the generalization ability of two methods. This initial point was not trained. The simultaneous method perform the task faster than sequential one.

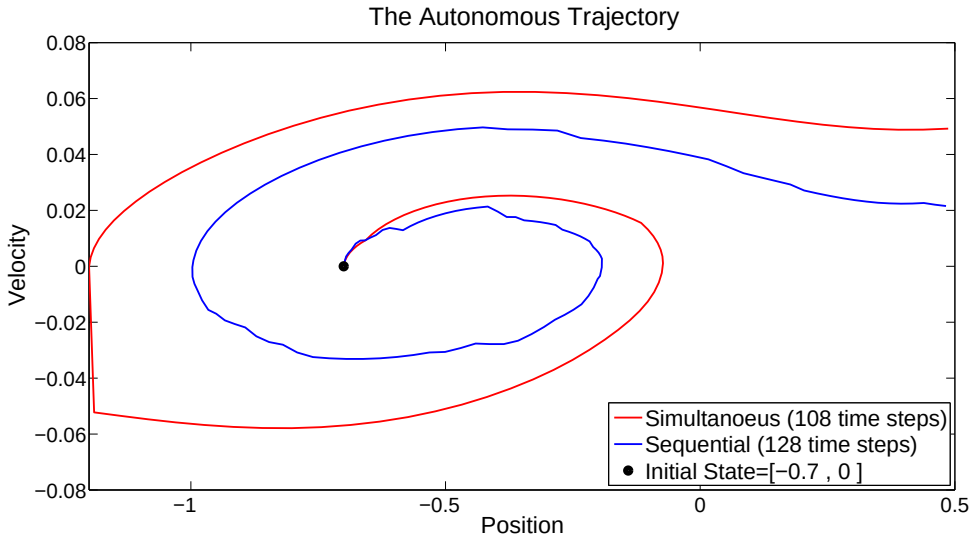


**Figure 51:** Comparison of the best Simultaneous run and the best Sequential run trajectories starting from the same point. This is testing the the ability of two methods in the given task. This initial point was trained. The simultaneous method perform the task faster than sequential one.

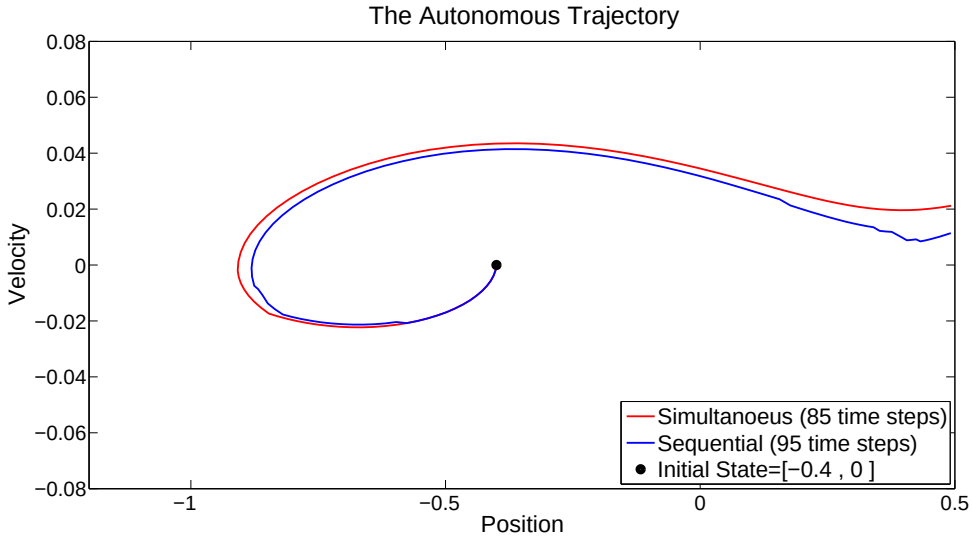


**Figure 52:** Comparison of the best Simultaneous run and the best Sequential run trajectories starting from the same point. This is testing the generalization ability of two methods. This initial point was not trained. The simultaneous method perform the task faster than sequential one.

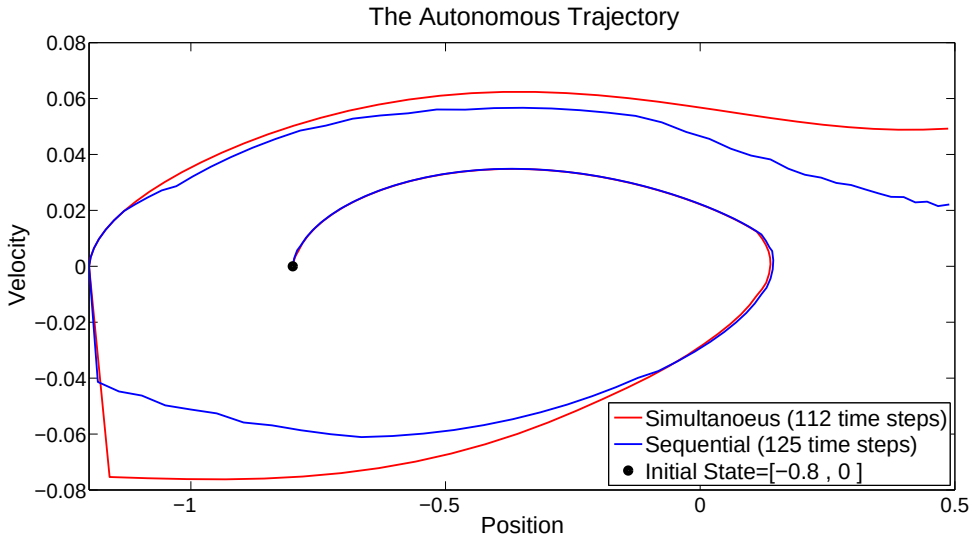


**Figure 53:** Comparison of the best Simultaneous run and the best Sequential run trajectories starting from the same point. This is testing the generalization ability of thwo methods. This initial point was not trained. The simultaneous method perform the task faster than sequential one.

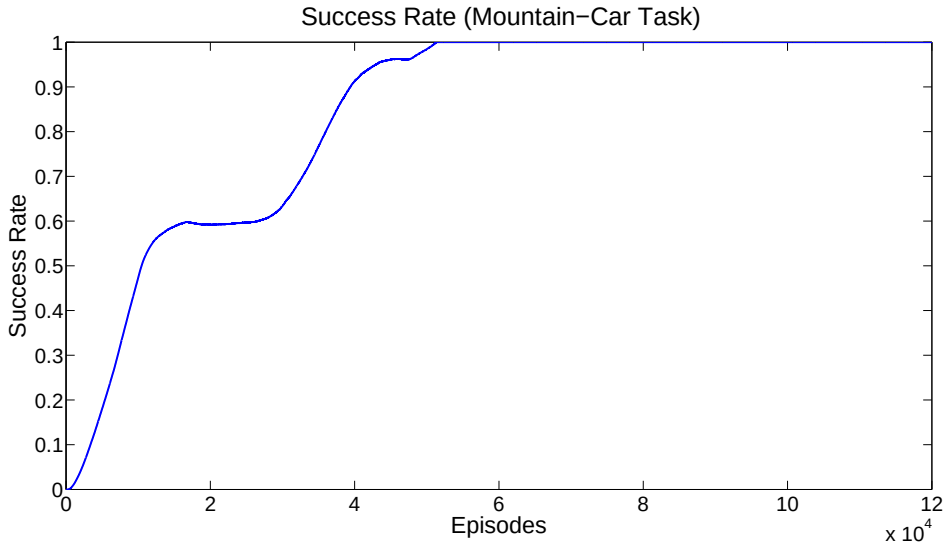


**Figure 54:** Success rate in the best run of the simultaneous method.

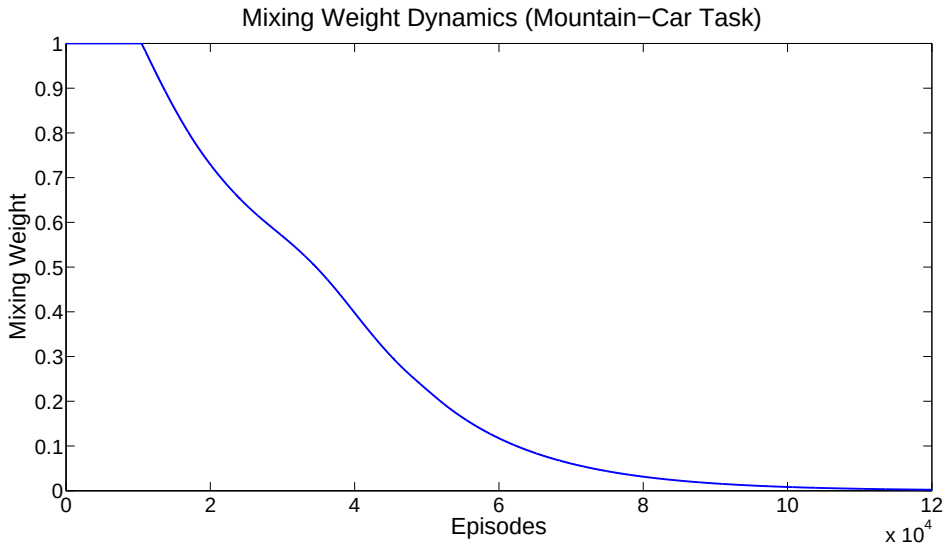


**Figure 55:** Mixing weight dynamics of the best run in the simultaneous method.

# CHAPTER V

# CONCLUSION

In this thesis, a simultaneous human-robot learning system is proposed for robot skill transfer. It is based on human-in-the-loop robot learning framework, and thus relies on a human operator to actively learn to complete a task through a robotic device. However, we addressed specifically the less studied topic of simultaneous control and learning of human and the robot. The novelty which is introduced over the existing methods are (1) state based control sharing, and (2) well defined dynamics of the mixing coefficient based on the overall performance, rather than the prediction error.

The proposed system was realized on a physical cart-pole system and also simulated cart-pole, speed-up task. It is shown to be effective in generating pole swing-up and balance controller in hardware as well as the mentioned simulation tasks. Additional experiments with naive subjects gave encouraging results that simultaneous learning can be more beneficial when the control mixing and learning regime are designed suitably. Intuitively, simultaneous learning can be understood as obtaining help from the robot which reflects the demonstrator's own policy of sometime in the past. In other words, the robot control helps to *correct* inappropriate actions in a given state based on the average of the correct actions that were issued in that state leading to good performance. So in this sense, we were expecting better autonomous policies from the simultaneous learning framework. The results in the cart-pole hardware part, although not fully conclusive are in this direction. A counter argument to this is the fact that the learning task for the human becomes more complex, as the controlled robot is not stationary in the simultaneous learning case. In fact when we asked the subjects to do both sequential and simultaneous trials (this is after

the experiments reported in this paper were finished) they stated that simultaneous robot control was harder. And, then the results of cart-pole simulation which is only focused on balance task (which is easier) show that simultaneous learning method is much more efficient than the sequential one.

In the naive subjects experiment parts, both groups were given the same amount of time to work with the cart-pole setup. However, the subjects in the simultaneous group had to deal with a harder task. It is likely that most subjects did not have enough time to reach their best during the cart-pole hardware experiments. If the goal is to obtain better policies the experiments should be continued until the success level of the subjects plateau. However, in the cart-pole balance task, by increasing the length of the pole, and also nature of the task itself, the difference between two methods become more clear.The simultaneous framework is also applied in speed-up task which was naturally different than two other task to show the potential of the framework.

The reinforcement learning (RL), as a simple model for human learning, is employed to have better understanding of the proposed framework. The mixing of reinforcement learning and machine learning (RL-ML) (representing simultaneous framework) had better performance in mountain-car task. It is also shown that the simultaneous method on average can does the task faster when it is started from different initial state which means it benefits from the generalization ability. Of course, the reinforcement learning's performance can be enhanced (e.g. by function approximation) and also it should be considered that exactly same RL used in both simultaneous and sequential methods. Then, this study is not comparing two RL-ML and RL. It gives an intuition of why human-robot simultaneous framework can perform the task efficiently.

There are several lines of future works. Firstly, the experiments should be conducted with more subjects, and the subjects must be given longer time to work.

Secondly, the state dependent weight dynamics must be based on an automatic state partitioning, or the dependence should be kept continuous. Thirdly, the open parameters of the framework, i.e. time constant for the weight dynamics and temporal window size for local success computation must be automatically adapted for each subject. Lastly, the framework should be validated on other robots and tasks, and a range of feedback interfaces must be tested for relaying back the level of competence of the machine control to the human operator.

# Bibliography


[1] R. V. Florian, "Correct equations for the dynamics of the cart-pole system," *Center for Cognitive and Neural Studies (Coneural), Romania*, 2007.

[2] M. A. Goodrich and A. C. Schultz, "Human-robot interaction: a survey," *Foundations and trends in human-computer interaction*, vol. 1, no. 3, pp. 203–275, 2007.

[3] S. Schaal and C. G. Atkeson, "Learning control in robotics," *Robotics & Automation Magazine, IEEE*, vol. 17, no. 2, pp. 20–29, 2010.

[4] J. Peters and S. Schaal, "Learning to control in operational space," *The International Journal of Robotics Research*, vol. 27, no. 2, pp. 197–212, 2008.

[5] L. Peternel and J. Babič, "Learning of compliant human–robot interaction using full-body haptic interface," *Advanced Robotics*, vol. 27, no. 13, pp. 1003–1012, 2013.

[6] J. Kober, E. Oztop, and J. Peters, "Reinforcement learning to adjust robot movements to new situations," in *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, vol. 22, p. 2650, 2011.

[7] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot programming by demonstration," in *Springer handbook of robotics*, pp. 1371–1394, Springer, 2008.

[8] L. Peternel and J. Babic, "Humanoid robot posture-control learning in real-time based on human sensorimotor learning ability," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pp. 5329–5334, IEEE, 2013.

[9] S. Schaal, "Is imitation learning the route to humanoid robots?," *Trends in cognitive sciences*, vol. 3, no. 6, pp. 233–242, 1999.

[10] C. G. Atkeson, J. G. Hale, F. E. Pollick, M. Riley, S. Kotosaka, S. Schaul, T. Shibata, G. Tevatia, A. Ude, S. Vijayakumar, *et al.*, "Using humanoid robots to study human behavior," *IEEE Intelligent Systems and their applications*, vol. 15, no. 4, pp. 46–56, 2000.

[11] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.

[12] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, vol. 1. MIT press Cambridge, 1998.

[13] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[14] J. Morimoto and K. Doya, "Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning," *Robotics and Autonomous Systems*, vol. 36, no. 1, pp. 37–51, 2001.

[15] E. Todorov, "Efficient computation of optimal actions," *Proceedings of the national academy of sciences*, vol. 106, no. 28, pp. 11478–11483, 2009.

[16] E. Theodorou, J. Buchli, and S. Schaal, "A generalized path integral control approach to reinforcement learning," *The Journal of Machine Learning Research*, vol. 11, pp. 3137–3181, 2010.

[17] M. Deisenroth and C. E. Rasmussen, "Pilco: A model-based and data-efficient approach to policy search," in *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pp. 465–472, 2011.

[18] S. Schaal, A. Ijspeert, and A. Billard, "Computational approaches to motor learning by imitation," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 358, no. 1431, pp. 537–547, 2003.

[19] J. Kober, A. Wilhelm, E. Oztop, and J. Peters, "Reinforcement learning to adjust parametrized motor primitives to new situations," *Autonomous Robots*, vol. 33, no. 4, pp. 361–379, 2012.

[20] D. Kushida, M. Nakamura, S. Goto, and N. Kyura, "Human direct teaching of industrial articulated robot arms based on force-free control," *Artificial Life and Robotics*, vol. 5, no. 1, pp. 26–32, 2001.

[21] J. Saunders, C. L. Nehaniv, K. Dautenhahn, and A. Alissandrakis, "Self-imitation and environmental scaffolding for robot teaching," *International Journal of Advanced Robotic Systems*, vol. 4, no. 1, pp. 109–124, 2007.

[22] J. Saunders, C. L. Nehaniv, and K. Dautenhahn, "Teaching robots by moulding behavior and scaffolding the environment," in *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pp. 118–125, ACM, 2006.

[23] B. Moore and E. Oztop, "Robotic grasping and manipulation through human visuomotor learning," *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 441–451, 2012.

[24] L. Peternel, T. Petrič, E. Oztop, and J. Babič, "Teaching robots to cooperate with humans in dynamic manipulation tasks based on multi-modal human-in-the-loop approach," *Autonomous robots*, vol. 36, no. 1-2, pp. 123–136, 2014.

[25] J. Babič, J. G. Hale, and E. Oztop, "Human sensorimotor learning for humanoid robot skill synthesis," *Adaptive Behavior*, p. 1059712311411112, 2011.

[26] E. Oztop, L.-H. Lin, M. Kawato, and G. Cheng, "Dexterous skills transfer by extending human body schema to a robotic hand," in *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*, pp. 82–87, IEEE, 2006.

[27] E. Oztop, L.-H. Lin, M. Kawato, and G. Cheng, "Extensive human training for robot skill synthesis: Validation on a robotic hand," in *Robotics and Automation, 2007 IEEE International Conference on*, pp. 1788–1793, IEEE, 2007.

[28] R. Chipalkatty and M. Egerstedt, "Human-in-the-loop: Terminal constraint receding horizon control with human inputs," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 2712–2717, IEEE, 2010.

[29] R. Chipalkatty, H. Daepp, M. Egerstedt, and W. Book, "Human-in-the-loop: Mpc for shared control of a quadruped rescue robot," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pp. 4556–4561, IEEE, 2011.

[30] C. Bringes, Y. Lin, Y. Sun, and R. Alqasemi, "Determining the benefit of human input in human-in-the-loop robotic systems," in *RO-MAN, 2013 IEEE*, pp. 210–215, IEEE, 2013.

[31] G. Ganesh, A. Takagi, R. Osu, T. Yoshioka, M. Kawato, and E. Burdet, "Two is better than one: Physical interactions improve motor performance in humans," *Scientific reports*, vol. 4, 2014.

[32] S. Schaal and C. G. Atkeson, "Constructive incremental learning from only local information," *Neural Computation*, vol. 10, no. 8, pp. 2047–2084, 1998.

[33] M. G. Azar, V. Gómez, and H. J. Kappen, "Dynamic policy programming," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 3207–3245, 2012.

[34] J. R. A. R. K. Mellon *et al.*, *How can the human mind occur in the physical universe?* Oxford University Press, 2007.

[35] C. B. Holroyd and M. G. Coles, "The neural basis of human error processing: reinforcement learning, dopamine, and the error-related negativity.," *Psychological review*, vol. 109, no. 4, p. 679, 2002.

[36] S. Vijayakumar and S. Schaal, "Locally weighted projection regression: Incremental real time learning in high dimensional space," in *Proceedings of the Seventeenth International Conference on Machine Learning*, pp. 1079–1086, Morgan Kaufmann Publishers Inc., 2000.

[37] M. A. Zamani and E. Oztop, "Simultaneous human-robot adaptation for effective skill transfer," in *Advanced Robotics (ICAR), 2015 IEEE International Conference on*, IEEE, 2015.