Robotic Manipulation with a Human in the Loop - The Pipeline for Learning by Demonstration

Sunil Srinivasan
Sebastian Schweigert
Jiewen Sun
James Su
Mark Jouppi

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2015-87
http://www.eecs.berkeley.edu/Pubs/TechRpts/2015/EECS-2015-87.html

May 14, 2015

Acknowledgement

Graduate student advisor: Aaron Bestick

University of California, Berkeley College of Engineering

MASTER OF ENGINEERING - SPRING 2015

Electrical Engineering and Computer Sciences

Robotics and Embedded Software

ROBOTIC MANIPULATION WITH A HUMAN IN THE LOOP – THE PIPELINE FOR LEARNING BY DEMONSTRATION

SUNIL SRINIVASAN

This **Masters Project Paper** fulfills the Master of Engineering degree requirement.

Approved by:

1. Capstone Project Advisor:

Signature: ____________________________ Date _____________

Print Name/Department: PROFESSOR RUZENA BAJCSY, ELECTRICAL ENGINEERING & COMPUTER SCIENCES

2. Faculty Committee Member #2:

Signature: ____________________________ Date _____________

Print Name/Department: PROFESSOR RONALD S. FEARING, ELECTRICAL ENGINEERING & COMPUTER SCIENCES

University of California, Berkeley College of Engineering

MASTER OF ENGINEERING - SPRING 2015

Electrical Engineering and Computer Sciences

Robotics and Embedded Software

ROBOTIC MANIPULATION WITH A HUMAN IN THE LOOP – THE PIPELINE FOR LEARNING BY DEMONSTRATION

SUNIL SRINIVASAN

ABSTRACT: Current practical-use robots are designed for very specific tasks and cannot be generalized to arbitrary tasks. However, to spread the use of generalized-task robots, they must be able to perform precise tasks, but must also be cost-effective; the high precision that single-task robots usually have is accompanied by a high hardware cost.

We propose a system that allows lower-cost robots to learn and attempt tasks that require high precision while failing safely, allowing the system to detecting portions of the task that it cannot complete alone, and then requesting human assistance for precision or accuracy that it cannot achieve. The system generates a model of the task from demonstrations, then analyzes the model to determine the accuracy required during different segments of the task. Subsequent comparison to the robot's known tolerance specifications allows the system to flag portions the robot cannot handle. Our approach generates the task model from a set of processed trajectories and a Gaussian mixture model, then extracts an ideal trajectory for the task as well as a set of standard deviations for each dimension of the trajectory data, allowing us to parametrize the accuracy for each segment of the task.

We also analyze industry factors to determine where such a system would fit in the current market for robotic technology, for a more comprehensive picture of where the technology stands.

Robotic Manipulation with a Human in the Loop

Sunil Srinivasan

Co-authors: Sebastian Schweigert, Jiewen Sun, James Su, Mark Jouppi

Final Capstone Report – The Pipeline for Learning by Demonstration

1 PROBLEM STATEMENT

(Authors: Schweigert, Srinivasan, Sun, Su, Jouppi)

We live in an age of increasing automation, but while we have machines that can open a can, pour a glass of water, or assemble a piece of furniture, the world does not have a machine that is versatile enough to do *all* of these tasks.

Normally, when people think of automation, they think of robots designed to accomplish a very specific task, such as lifting a car-door into a car-frame, over and over again. The newer generation of robots though is the class of general-purpose robots. While such robots have yet to materialize commercially, general-purpose is a great concept. Imagine if families could have a robotic assistant to take care of household tasks or run daily errands. In short, human life would be much more convenient and efficient.

Unfortunately, a major limitation towards reaching such a milestone is the engineering trade-off between cost and performance: with a limited budget of resources, it is almost impossible to add additional levels of complexity without decreasing performance. As such, it has traditionally been challenging to use robots that are both low-cost and versatile in domestic environments because the applications of these robots are limited by their low performance – specifically, their inaccuracy. This is where we decided that the human should come in.

To address this barrier of limited resources, our capstone team has developed a system that is designed around robot-human interaction, where human instructors train and work with cost-effective robots to accomplish a broad range of tasks with high accuracy. Using a set of algorithms that we have developed, the robot learns how to perform a task from a human who teaches it through a series of demonstrations. Following this learning process, the robot evaluates the task and identifies the precision requirements using a mathematical model. And when the robot detects that it is unable to achieve the accuracy required for a certain portion of the task, it requests human assistance. The final outcome is a system that excels across a vast range of duties, due to the combination of both the efficiency of robots working on a large-scale and the precision of humans working on a small-scale.

This revolutionary design of cooperation between man and machine succeeds at tasks that are otherwise impossible for the machine to accomplish alone. In essence, we added in the human as an additional resource to improve the overall performance of the system. This was the rationale behind our capstone project, for we saw an opportunity here to make an enormous technical stride in society's current usage of commercial robots: we took an otherwise unimpressive commodity – the low-cost and inaccurate robot – and engineered commercial value from it in the form of robotic adaptability.

2 INDUSTRY AND MARKET TRENDS

(Authors: Schweigert, Srinivasan, Sun, Su, Jouppi)

Before examining any technical details though, we first wanted to scope out the business potential of our project. Consequently, in an attempt to analyze our strategic position in the market, we evaluated the competitive forces outlined by Porter (Porter, 2008) because we felt that an in-depth analysis of the intensity of these forces will influence our marketing strategies. In other words, analyzing these five forces enabled us to have a better understanding of our industry and shaped our strategies to sustain long-term profitability. Before we begin our analysis however, let us first clearly define both our market and our product.

2.1 Market and Product

We defined our market to be consumer households with the intent that our algorithms accomplish household tasks, such as assembling furniture. We chose to target the elderly and the disabled as buyers of our product because this is a large, growing population with critical and largely unmet needs. Simply put, the elderly population in the United States is growing. While the current number of senior citizens in the US is roughly 40 million, that number is expected to grow to over 80 million by 2050 (Ortman et al., 2014). Additionally, according to US 2010 census data, about 30%, 38%, and 56% of the population aged 70 to 74, 75 to 79, and 80 or over, respectively, live with severe disabilities (Brault, 2012). To further narrow our market though, we chose to focus specifically on affluent elderly-and-disabled individuals as our target customers. This is a reasonable objective because many elderly people have amassed a wealth savings and investments cultivated over their lifetimes. Indeed, according to a Pew Research study, the median net worth of senior citizens in the US is $170,000, which is 47 times greater than that of people aged 35 and younger (Censky, 2011).

The definition of our product is a more complex matter because, at its core, our capstone project involved the research and development of an algorithm that allows a robot to learn a task and cooperate with a human to perform that task; it is not a complete software – or hardware – solution. Unfortunately, while software solutions usually have commercialization potential, algorithms alone do not. In order to take our robot-learning algorithm and relate it to a commercial application, we had to decide what form that application should take and how to take such a product to market. One option was to simply license out our algorithm for others to utilize; we would receive royalties as a result of these sold licenses, and companies could make products or provide services using our algorithm. One major caveat, though, is that our algorithm incorporates ideas presented in externally-published research, so the intellectual property for this method may not lie entirely with us. We therefore chose not to investigate this option any further. Our next option to consider was to sell a software solution for users to install on devices that they already own. However, the “device” in this case would be a full-fledged robot, where, as a point of reference, a Baxter robot from Rethink Robotics – our current hardware-platform of choice – has a set price of approximately $35,000 (Rethink Robotics, 2015). Clearly, it would be ludicrous for people to purchase such costly technology without ensuring that it already comes with the necessary software to function. This left us with our final choice: a “full package”, in which we offer a robotic apparatus preloaded and set up with our software such that a consumer only needs to buy one product, with installation services if necessary. This way, we can market our product directly to our target consumers and eliminate the customer’s barrier-to-purchase that comes from setting up the technology. Thus, we decided on this “full package” as the form for our product: a physical robot bundled with software algorithms that we implement.

We must consider several factors with the decision to market this “full package”. The first is price, and this is largely influenced by the suppliers since we must obtain the proper robotic hardware or components externally. After all, according to an IBISWorld report, the cost of mechanical manufacturing is increasing as the expenditure of raw materials increases, so we opt to purchase a whole robot setup instead of building our own robot from basic components (Crompton, 2014:25). As a result, we would look to Rethink Robotics as a supplier of our Baxter robot, a hardware platform. With a markup from our software and services, selling our product at around $40,000, or at about a 15% markup, is not an unreasonable price point – especially if we were to get an Original Equipment Manufacturer (OEM) discount for Baxter. This provides us with a defined pricing model.

Lastly, we must discuss promotion and place/distribution. As O'Donnell points out, 50% of seniors are not using the internet, so marketing is better achieved through conventional channels such as mail, television, and newspapers (O'Donnell, 2015). Interestingly, O'Donnell also predicts an increased use of social media by seniors in 2020, making social-media campaigns a possibility in the near future (O'Donnell, 2015). Distribution of this product, however, is complicated; while we would like to be able to sell our product online, providing setup services would require a trained professional to be present. As such, we will most likely have to either distribute through local partners that provide such services or create a local presence ourselves, incurring additional costs. With our product, price, promotion, and place now defined, we have all the significant facets of a commercialization strategy. Note that we do not analyze the minimum viable product (MVP) in detail. This is because our research specifically investigates the Baxter robot's ability to learn the task of assembling a coffee table, at which point we will have a decent MVP that performs table assembly. Thus, we have established a viable (if hypothetical) commercialization strategy for our research efforts.

2.2 Competitive Forces Analysis

2.2.1 Power of Buyers

With the market and product definition out of the way, we can begin to evaluate Porter's five forces, the first of which is the power of buyers (Porter, 2008). We deduce this force to be relatively weak, since the large population of potential buyers means that individual buyers do not have much leverage or bargaining power with us in our product offering. Moreover, as we will address later on, there are few – if any – direct rivals in our industry. Thus, a scarcity of competing products only elevates our power, as options are limited for the buyer. Furthermore, the switching costs for complex, robotic solutions would be high; given that the price of these robots with our software would be roughly $40,000, it is not an expense to be made frequently. We imagine that a typical customer will only purchase one such robotic system in their life. Thus, it is not of great concern that customers would switch to using a competitor's domestic robot solution after purchasing our product. All in all, the power of buyers is assessed to be fairly weak, and we do not concern ourselves in mitigating this force.

2.2.2 Power of Competitors

Regarding rivalry within our industry, there are two main classifications of competitors: robotics companies and robotics research institutions. Some of these competitors offer products that are mildly similar to our envisioned product, and they also target similar markets. For example, Clearpath Robotics, a robotics company (Hoover's, Inc. "Clearpath Robotics Inc.," n.d.), offers support to the PR2 robot to perform household chores like fetching food from the refrigerator and cooking it. Alternatively, there are research institutions like the Information and Robot Technology Research Initiative (IRT) at the University of Tokyo working on developing software that allows the AR robot to accomplish household assignments such as cleaning floors, washing laundry, and so on. Fortunately, companies and research institutions like these will only indirectly compete with us because our product differs from theirs in the extent that humans are involved. The robotic systems these competitors are developing are meant to be fully autonomous – the robots execute their tasks independent of any human interaction – while our system is meant to be semi-autonomous, enabling a human to both work with and teach a robot to perform various tasks. This is an advantageously superior method because now the scope of the system is not limited to what the robot can accomplish independently; the scope is broadened to what the robot and human can accomplish together synergistically. Simply put, the generality of our method enhances a robot's utility and flexibility. Apart from offering a unique product though, we also have some advantages over our competitors in terms of hardware costs. To illustrate, a two-arm PR2 robot is priced around $400,000 (Owano, 2011) while a Baxter robot, as mentioned previously, is priced around only $35,000 (Rethink Robotics, 2015), a relatively far-cheaper option. To summarize, since we are working in a fairly new field, there are no true established rivals in this specific area yet. Thus, we can conclude that the force of competitors is weak.

2.2.3 Power of Substitutes

Moving onto the next force listed by Porter, we realize that significant attention needs to be given to the force of substitutes since there are, broadly speaking, quite a number of substitutes to our product. For instance, alternative technologies, like the iRobot Roomba (Biesada, n.d.) – a popular floor cleaning robot, have existed in the consumer market for many years, and these established technologies have a large customer base. Customers are more comfortable with familiar products, so it will not be easy to encourage customers to migrate to a substitute product. Moreover, if we look past the technological substitutes, there are a variety of human-labor alternatives in regards to accomplishing household tasks, such as employing a live-in caretaker or residing in a nursing home. However, similar to our stance against the competitor force, we again have some advantages due to our functionality and low cost. Addressing the concern of alternative technologies, even though products like the iRobot Roomba are popular and functional, they tend to have a limited set of features, such as floor cleaning. Our product, on the other hand, is a more general solution which can be used to tackle a variety of household chores. Along that same line, for many tasks in this set, our robot can be more efficient than a human caretaker due to its autonomous nature. Furthermore, as mentioned previously, our pricing model markets our product at a cost of about $40,000 with an extensive lifespan, while most nursing homes cost up to $80,000 – and that is per year (Ellis, 2013). All of these arguments make our product competitive to existing substitutes, motivating us to divert attention from this force and concentrate on more pressing ones.

2.2.4 Power of New Entrants

In contrast to the mild nature of the forces mentioned previously, new-entrant competition looming over the horizon should be of great concern. For instance, some of the heavy-hitters in robotic research include companies like Clearpath Robotics (McMillan, 2015) and 3D Robotics (Bi, 2015), both of which were founded only six years ago in 2009. It seems that, unlike the issue of existing rivals and possible substitutes, there is indeed a strong force in regards to new entrants. To further illustrate this fact, large corporations with broader goals in the technological field can certainly seep into our industry, such as Amazon with its Amazon Prime Air drones or Google with its autonomous cars. Big players such as these would certainly have the resources to quickly create a new division within their company and fund research in alternative robotic avenues. Furthermore, even our suppliers can be considered possible new entrants, since they both already posses their own hardware and can additionally reverse-engineer our software algorithm that was, in large-part, acquired from public research papers. All in all, to summarize, we see that dangerous incoming players in this industry are: either startups or big companies with other additional focuses, or suppliers that provide our hardware. When combined with the fact that there are no true established rivals yet as mentioned previously, this danger reinforces both the notion that robotics is a relatively new field and that the threat of new entrants is high.

2.2.5 Power of Suppliers

The last of Porter’s five forces to address is the threat of suppliers (Porter, 2008). This threat is a complex point that requires careful analysis in our business strategy. To first clarify, we envision robotic-hardware-platform manufacturers as our suppliers. As per our product description, we would take the robotic hardware platforms from companies like Rethink Robotics and Universal Robotics, customize the robots with our specialized software that gives them practical intelligence to work alongside humans, and then sell them to customers. In particular, we would purchase from companies that produce innovative, low-cost robotic hardware platforms upon which we can then build our solution. Our smart software would make up for the inaccuracies in the cheaper hardware with better algorithms and human-in-the-loop collaboration. Since there are currently only a few firms producing such low-cost platforms, these few suppliers have high bargaining power, as we are left with fewer alternate firms from which to choose.

2.3 Market Strategy

We see that presently, of Porter’s five forces, both new entrants and existing suppliers hold the most power (Porter, 2008). Knowing this, we can establish our market strategy to mitigate these two forces, strategically positioning ourselves in a superior situation.

To mitigate the threat of new entrants from the suppliers themselves (see Section 2.2.4), we can generalize our software to work across multiple platforms and disincentivize suppliers to enter the market, as they would only be encouraged to produce software across their own single platform. Additionally, to discourage new and small startups from forming, we can both establish strong relationships with suppliers to gain a leg up on others looking to pursue our method of utilizing existing hardware and maintain a high fixed cost – such as a high R&D cost by developing more proprietary algorithms – to deter incomers that have a small amount of seed funding. Finally, we can address the threat of entry from large corporations by realizing that these companies have more overarching goals, so focus on their robotics branch will not be as heavy as on their other branches. As such, we can capture a niche market to detour focus and attention away from us. Fortunately, we have already positioned ourselves in such a situation, in which we target a niche group of customers – the elderly and the disabled. As a result, we see that our competitive landscape as it applies to new entrants can be classified as quite aggressive, but there are indeed routes we can take to dodge much of this aggression.

To mitigate the issue of being locked into a single supplier (see Section 2.2.5), the core strategy is still to generalize our software. This would considerably increase our power, since we would no longer be dependent on any one supplier. Note that as a trend, robotics startups are becoming increasingly common (Tobe, 2013), and we thus anticipate more suppliers coming into the market in the future. As of right now though, suppliers are a strong force that must be considered carefully in our strategy, and we must route efforts to ease this force.

2.4 Market Trends

With an evaluation of competitive forces complete, we end with a discussion of the major trends that influence our project strategy. Aside from the trends of both the changing age demographic of the US – affecting the power of both buyers and substitutes – and the increased interest in the robotic industry – affecting the power of both substitutes and new entrants, another trend to consider is the recent advancement in integrated circuit (IC) technology that has resulted in improved computing performance and reduced cost, resulting in reduced barriers-to-entry and thus further enhancing the threat of new entrants. IC technology has seen consistent improvements in computing power and consistent reductions in cost since their inception. Our industry is directly affected by these advancements; in recent years, more powerful computational devices have generated more robotic technology in the household arena, for engineers are allowed to easily incorporate computing power into the chassis of the robot. This design contrasts with industrial robots, where the computational power is often located in an external computer. The trend is summarized with a concept known as Moore's law, stating that the computational power of the average IC doubles nearly every two years. This trend has been relatively consistent since the early history of ICs. However, there is disagreement among analysts about how much longer this trend will continue (Hoover's, Inc. "Home Health Care Services.," n.d.). The trend has the effect of making our products more functionally efficient and versatile, which reduces the power of substitutes. However, the lower cost of computing technology also reduces the barriers-to-entry in the industry, which increases the power of rivals. Only time will reveal the overall impact that this trend will have.

To summarize, from our strategy analysis, we have deduced that while some competitive forces are certainly in our favor, a few forces bring cause-for-concern and need to be addressed. With adequate industry analysis, we can plan our strategy in order to leverage ourselves into a better position within the market. Summarizing our findings, we have identified within the market both the power of new entrants and the power of suppliers to be strong forces. Consequently, to dampen these threats, we would generalize our software to work across multiple platforms, disincentivizing suppliers from entering the market as well as taking away supplier bargaining power. We would also encourage people to use our product instead of substitutes by having features and functionality that other products do not, at a price point that is not prohibitively expensive.

3 IP STRATEGY

(Authors: Schweigert, Srinivasan, Sun, Su, Jouppi)

Aside from a business standpoint though, we must also consider which legal avenues to take in order to protect our intellectual property (IP): in particular, whether or not our idea is patentable. After all, in many research scenarios such as ours, a patent is the most feasible way to safeguard any IP that is developed. Unfortunately, as this section will argue, patenting our work may not be the most practical path to pursue; however, we do have an alternative strategy better suited to our purposes, in the form of copyright.

We feel that in our more specific situation, the costs of attempting to obtain and enforce a patent far outweigh the benefits, for a number of reasons. One consideration is that the mathematics behind the algorithms we employ are pulled from published research papers, particularly those that deal with robot learning-by-demonstration (Billard et al., 2008). Therefore, the proprietary essence of such research is not ours to claim. By the same token, we cannot patent the ROS (Robot Operating System) software platform upon which we develop because it is open-source and thus, once again, publically available. Most importantly, we do not feel that it is pragmatic to patent the software code itself. This is because software, at its core, is the manifestation of logical deductions, and another group or individual may take a different route of logical deductions to arrive at the same conclusion. Following this train of thought, it is ordinarily quite difficult to obtain and/or protect a patent when the end result can be reached in various ways. As explained by Anthony Klein, an IP attorney at Latham & Watkins LLP, pure software patents remain controversial since “what would constitute patentable subject matter is unclear” (Klein, 2015).

Before investigating an alternative means at protecting our ideas however, it is important that we have the foresight to research whether existing patents overlap with our results. We discovered that the closest patent to our project is entitled: “Method and system for training a robot using human-assisted task demonstration” (Bajaras, 2014). It describes a system for humans to train robots for pick-and-place tasks by moving the robot’s arm while recording trajectory-and-perception data through the robot’s sensory systems. At first glance, it may appear that our project directly infringes upon this patent. However, after delving into the details, this is not the case due to the limited scope of this patent. To give some background on the nature of patents, a patent consists of independent claims and dependent claims; if one does not violate the independent claims, then by definition, one does not violate the dependent claims (Brown & Michaels, PC 2006). Now, many of our project’s similarities with this patent lie in the dependent claims. However, if we can argue that our capstone project does not infringe upon any of the independent claims, then we can legally claim that we do not infringe upon the dependent claims as well – and thus the patent as a whole.

There are two independent claims mentioned in this patent. To quote the first independent claim (claim 1):

> A method for training a robot to execute a robotic task in a work environment, the method comprising: moving the robot across its configuration space … assigning, via the ECU, virtual deictic markers to the detected perceptual features (Bajaras, 2014).

We argue that we do not infringe this claim because our project does not use “virtual deictic markers” – markers are based on a representational paradigm that use “selective attention and pointers … to learn and reason about rich complex environments” (Ravindran, 2007). As for the second independent patent claim (claim 2):

> The method of claim 1, wherein moving the robot across its configuration space includes moving at least one of a robot arm and a robot manipulator attached to the robot arm (Bajaras, 2014).

Our project does not use a single “arm and a manipulator”, but rather a dual-armed Baxter-robot. Hence, our project does not violate any of the independent claims and thus none of the dependent claims. Therefore, while this is the closest patent to our idea, we do not infringe upon it and are therefore not required to license from it. Since other existing patents are even less related, a breach of IP is of no worry to us.

With the threat of similar, existing IP out of the way, we can now begin to pursue an alternative strategy of IP protection. After much consideration, we believe that copyright is the most appropriate option – in fact, this happens to be the choice for many software companies. Of course, copyright does indeed present a few risks since, in general, patents protect ideas while copyright only protects the expressions of ideas.

The first risk is the risk of knock-offs: there are ways around copyright such that people can make products very similar to ours but are not in violation of copyright law. This includes implementing our algorithm through a different tactic – one example is converting our code to a different programming language – as well as merely adapting functions from our program. The point is that copyright does not protect our ideas, making it incredibly easy for others to take our ideas and tweak them to look slightly different in their end product. We would need to mitigate this issue by implementing our algorithm across multiple programming languages to prevent the scenarios where someone claims credit on our ideas based on simple modifications.

The second risk is the risk of undetected duplication. It is the first risk in reverse, where certain competitors are indeed copying our code directly, but we have no way of detecting that they are doing so. The reason for this is that we will generally not have the source code of our competitors to compare to our own; all we will have is the compiled functionality that their code is capable of demonstrating. In that sense, it is near impossible to identify specifically if they have violated copyright. Consequently, it is quite difficult to mitigate this risk.

While copyright does offer less protection than patents, it is nonetheless more feasible and realistic to acquire. For instance, copyright is granted automatically when an original work is created, so registration is not required. This simplified procedure immediately eliminates the time and money that we would otherwise need to spend to obtain a patent. Moreover, the duration of copyright is the life of the author(s) plus 70 years, which is plenty of time for us given the short life cycle of software. Furthermore, copyright offers authors the exclusive rights to reproduction, protects against public displays and derivatives of their work, and establishes a public credibility that can attract investment and customers. Licensing can also present itself as a way to increase profit and expand a business.

All in all, it appears that pursuing a patent is not the route for us to go. Instead, a more practical approach at protecting our IP is for us to pursue copyright, due to both the more lenient restrictions and more efficient timeline at obtaining copyright.

4 SYSTEM WORKFLOW

(Authors: Schweigert, Srinivasan, Sun, Su, Jouppi)

Shifting gears now to the technical details of our capstone project, let us clarify once more that we developed a general-purpose robotic system that incorporates robot-human interactions. Yet it is difficult to implement generality without first implementing and testing lower-level components. Therefore, as a starting point, we defined a specific task that we aimed to accomplish: having Baxter assembling a coffee table with the help of human. Having such an aim gave us a concrete goal for an implementation and test framework for our system (Fig. 1 shows the workflow of this system).

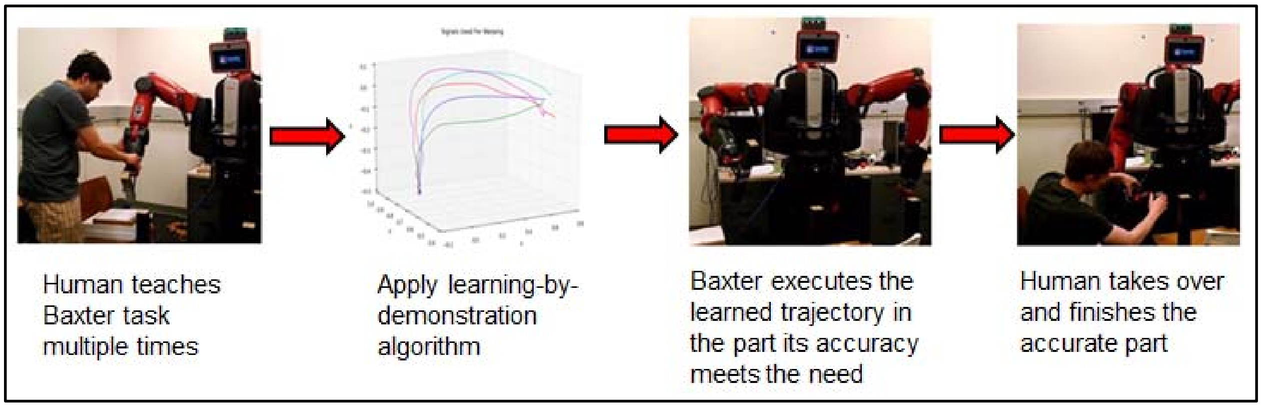
Human teaches Baxter task multiple times
Apply learning-by-demonstration algorithm
Baxter executes the learned trajectory in the part its accuracy meets the need
Human takes over and finishes the accurate part

Figure 1: Workflow of System Process

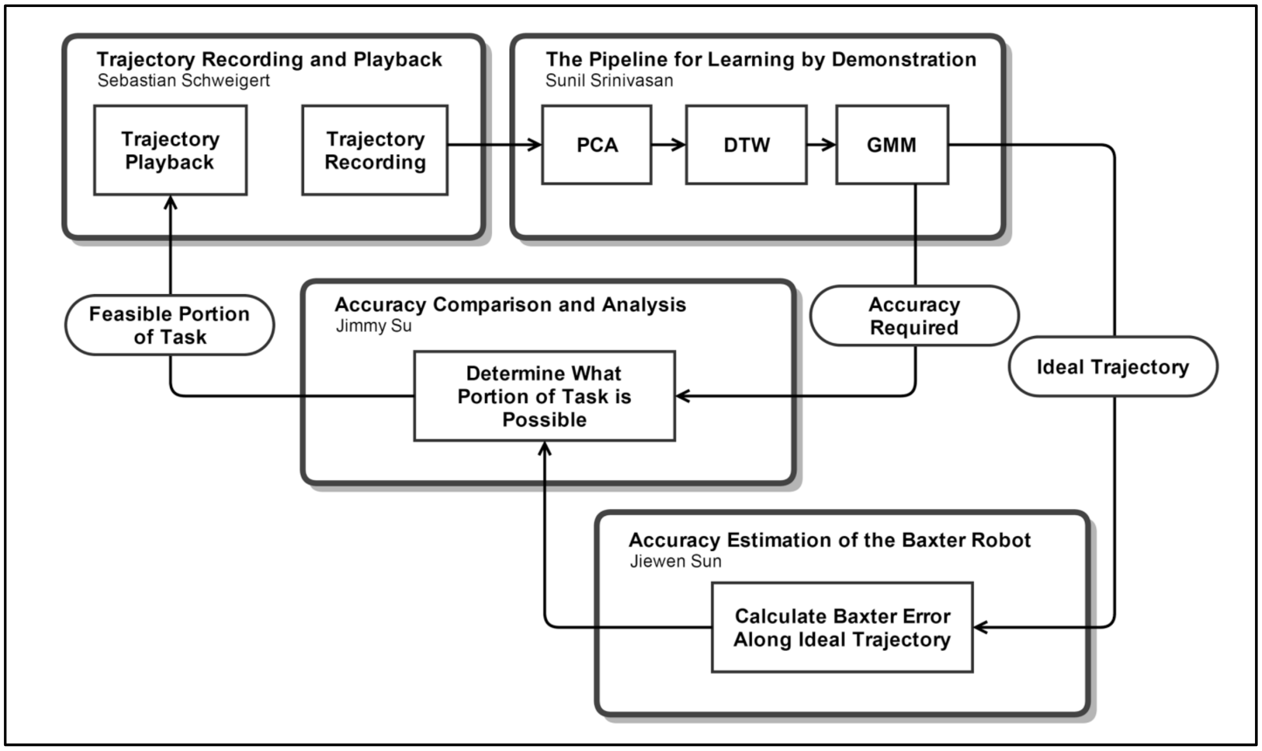
Trajectory Recording and Playback
Sebastian Schweigert
Trajectory Playback
Trajectory Recording
The Pipeline for Learning by Demonstration
Sunil Srinivasan
PCA
DTW
GMM
Feasible Portion of Task
Accuracy Comparison and Analysis
Jimmy Su
Determine What Portion of Task is Possible
Accuracy Required
Ideal Trajectory
Accuracy Estimation of the Baxter Robot
Jiewen Sun
Calculate Baxter Error Along Ideal Trajectory

Figure 2: System Component Organization

5 TECHNICAL CONTRIBUTIONS

(Author: Srinivasan)

5.1 Overview

Our capstone project is rooted in the idea that a low-cost robot, when performing or learning a task, should be able to infer the accuracy needed to perform that task. With that in mind, we focus on training the Baxter robot to learn the specific task of transporting and positioning a table leg for the assembly of a coffee table. Baxter, a two-armed robot with 7 degrees of freedom on each arm, costs much less than most robots with similar features; this is due to the inclusion of less expensive parts such as plastic gears (Rethink Robotics, 2015). With these parts come inaccuracies of motion, and our goal is to develop a software system that allows Baxter to learn the task, determine the accuracy and precision required to complete various sections of the task, and allow a human to assist it in completing the sections it cannot handle alone.

To complete this system, we divided our efforts into several key tasks. First, we need functionality for recording the trajectory data for the robot's arm motions and playing back that data as physical arm motions. This is fundamental to training a robot to both observe and perform a task, as we will be physically moving the Baxter arm to teach it a task. Next, we must generate a representation or model of the task from a number of recorded demonstrations of the task; this is the *learning by demonstration* method. Third, we must find a way to determine Baxter's inherent accuracy bounds given the current angles of the arm joints; we call this *accuracy parametrization*. Lastly, we must merge the task model and Baxter's accuracy bounds so that we can compare the accuracy specifications of the two. This set of features allows the system to determine whether the accuracy needed for the learned task is within the capabilities of the robot.

To divide these tasks among ourselves, we first split the tasks into those that worked with the robot's data and those that could be abstracted from that data. As such, portions of our team worked on each category, with the trajectory recording/playback and accuracy parametrization handled by one group and the learning algorithm handled by the other. My personal contributions included all the required implementation steps for the learning algorithm, such that we could input a set of recorded trajectories into a function and retrieve as output a model of the task represented by those trajectories.

5.2 Dynamic Time Warping

5.2.1 Literature Review

Initially, we decided on a naïve learning algorithm for preliminary testing that involved *dynamic time warping* (DTW) to align signals temporally, followed by a standard mean of those signals to generate a learned trajectory. We chose this because we reasoned that, before averaging any recorded trajectories, such trajectories must have the same time scale. DTW determines the optimal warping for two signals such that the resulting signals' shapes match as closely as possible according to some cost function, essentially approximating the same time scale. The algorithm computes a cost matrix between the two signals and finds the lowest-cost path through this matrix using dynamic programming. The result is a sequence of correspondences (or 'pairs') that map each sample in one signal to one or more samples in the other signal, and three conditions hold for this sequence. First, the first and last correspondences are always the first samples from each signal and the last samples from each signal, respectively. Second, the sample 'indices' for both signals increase monotonically in the correspondence. Lastly, the indices only ever increase by at most 1 for each signal, so that neighboring samples in one signal map either to neighboring samples in the other signal, or to the same sample in the other signal (Müller, 2007, p. 70). These conditions ensure that DTW preserves the ordering of samples and ensures that all samples are present when warping the signals.

The dynamic programming implementation for DTW is fairly involved: first, it calculates an accumulated cost matrix by iterating through every pair of samples from both signals and setting the pair's accumulated cost based on the distance between the samples and the minimum cost of its predecessor pairs. Predecessor pairs are made up of either the previous samples from both signals, or from the previous sample of either signal paired with the current sample of the other signal. Then, the minimum path is calculated by starting from the end of both signals and traversing the accumulated cost matrix backwards, finding the predecessor pairs that have the least accumulated cost (Müller, 2007, p. 72-73). The resulting path defines the sequence of correspondences for warping the signals.

5.2.2 Methods and Design

My Python implementation of the DTW algorithm differs slightly from that outlined by Müller. For one, I performed some bookkeeping steps during calculation of the accumulated cost matrix that allow me to skip the backwards traversal and generate the correspondences directly from my bookkeeping results. In addition, instead of warping two signals to match each other, I fixed one signal as a reference point and warped the other signal to match this reference signal. As a result, if one sample in a warped signal maps to multiple samples in the reference signal, my implementation takes the arithmetic mean of those samples to get the warped signal value at that time. Lastly, there are several cost functions available for calculating the ‘distance’ between two samples of the two signals; I chose to use the Euclidean distance, which takes the element-wise difference between the two samples, then squares every element and takes the square root of the sum of those squares. I added the option to specify an alternative cost function, in case the Euclidean distance is not appropriate for the data, as well as a locality constraint window, which prevents warping from assigning too many samples in one signal to a single sample of the other signal (the warping path is forced to stay within the window).

5.2.3 Results and Discussion

There are two outcomes of these efforts. The first is a flexible function that takes as input two signals of arbitrary length and dimensionality and produces as output two correspondence arrays that map the signal samples to one another. The second is a helper function that uses the correspondences from the first function to shift the samples of the second signal (in time) to minimize the distance from each sample to the sample with the same time value in the first signal (which is the reference signal). Fig. 3 provides an example of what the function does, as run on two-dimensional test inputs with Signal 1 as the reference signal. Note that the peaks and troughs of the two signals align in the second plot, indicating that the time values of Signal 2's samples have been modified to match Signal 2 as closely as possible with Signal 1's samples. In particular, the samples that make up the rising and falling edges of Signal 2 have shifted to match the rise and fall of Signal 1's value more closely, and the other samples have shifted to account for this. With respect to the project as a whole, the trajectory data from the trajectory recording task stores the signal as a large matrix with samples as rows and data dimensions as columns, allowing it to be input into the function (Schweigert, 2015). For a set of trajectories, we can pick one as a reference signal and warp the others to align with that signal; all four then have the same number of samples and the same time values. Early in the project, we used this to make Baxter ‘learn’ an average trajectory made up of the mean of temporally-aligned trajectories.

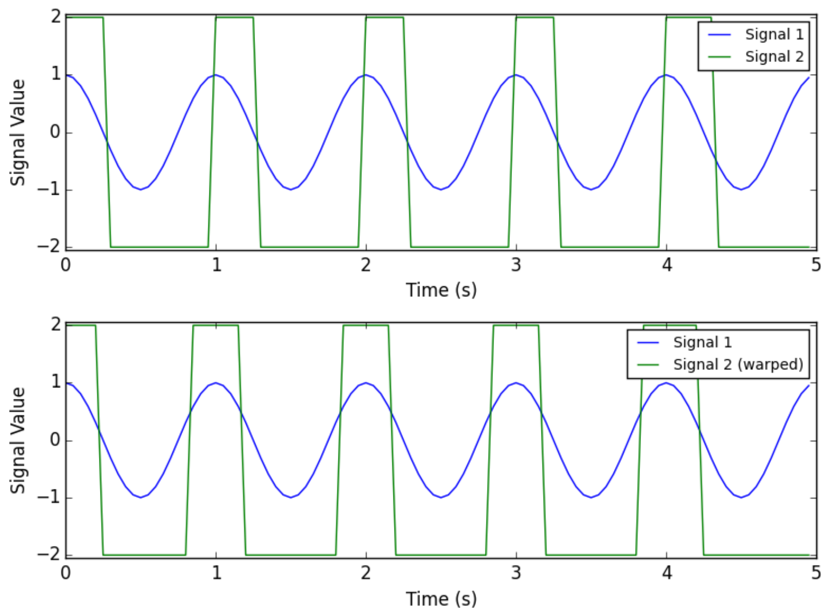

Figure 3: DTW output for example 2D signal.
The first plot displays the original signals.
The second plot displays the original Signal 1, as well as Signal 2 after DTW.
Note that the samples of Signal 2 have been shifted such that the
peaks and troughs of Signal 1 and the warped Signal 2 now align.

5.3 Learning by Demonstration

5.3.1 Literature Review

After implementing and testing the DTW algorithm, it was clear that a naïve mean of trajectories would work for simple, smooth motions, but would not be able to model tasks that are more complex. Moreover, simply capturing the mean of a set of trajectories did not provide us with a clear way of determining the accuracy that a task required. As such, we began to look into further algorithms for robot learning by demonstration. As a part of this literature review, our examination included a look at Abbeel and Ng's (2004) work on apprenticeship learning. They lay out a situation in which the goal of the learning algorithm is to solve a Markov decision process that lacks a reward function, essentially being given only the available actions, the state transition probabilities, and the feature expectation for some set of features that determine the (unknown) reward value. The given feature expectation is that of an expert performing some task, such that they call this the *apprenticeship learning algorithm,* and the algorithm attempts to find a policy of action such that the performance of the policy is similar to that of the expert despite the reward function being unknown. Essentially, the apprenticeship algorithm attempts to come up with some reward function for which the expert performs well, then creates a policy that performs well for that reward function (Abbeel & Ng, 2004, p. 3). Regardless of the 'format' of the behavior, then, as long as the Markov decision process is laid out properly, the returned policy should theoretically mimic the expert's behavior. However, we decided not to implement this algorithm for use in our project, mainly because formulating our project's problem as a Markov decision process would be overly complicated for our needs. An additional issue involves the fact that the algorithm generates an ideal policy for the robot's actions, but it does not provide any kind of metric for accuracy, which is what we are mainly interested in. Thus, we moved on to literature that dealt with accuracy metrics as a part of learning by demonstration.

We then turned our attention to Calinon, Guenter, and Billard (2007), who lay out a strategy for generalizing a task using a *Gaussian mixture model* (GMM) for generation of a *learned trajectory*. The tasks they lay out involve robot interaction with some object at some arbitrary position in the world space, such that the situation they set up is very similar to our own. They detail a pipeline that processes a set of trajectories into a set of features of a task, which can then be queried at increasing time steps to generate a generalized trajectory for the task. We focused on their algorithm in particular from the body of work because the Gaussian mixture model's variances provide an inherent metric for accuracy needed during a task.

Their pipeline begins with a particular representation of the trajectory data. First, they note the initial positions of objects in the world in a preliminary phase. Then, they record the joint angles of the robot arms, the Cartesian positions of the robot's hands, the relative Cartesian positions of the hands to the objects in the world, and the status of the hands as being open or closed, appending a time value to each point (Calinon et al., 2007, p. 288). The first three data sets (joint angles, hand positions, and relative hand positions) are in relatively high-dimensional space, as Cartesian space for two robot hands results in 6 dimensions and the number of dimensions for joint angles depends on the number of joints in the robot's arms; this results in less optimal processing and some degree of redundancy. To make the problem simpler, Calinon et al. subtract the mean value of each dimension, then apply Principal Component Analysis (PCA) to reduce the dimensionality of each data set from the *original data space* into a *latent space* (note that they omit the variable of time for this). This also causes the dimensions in the latent space to be uncorrelated, and they decide to retain enough dimensions such that 98% of the original data's spread is covered by the latent space (Calinon et al., 2007, p. 289).

Next, Calinon et al. (2007) address the issue that the various signals are not temporally aligned. While they mention that their previous work used hidden Markov models to handle temporal variations, they opt to use DTW here to align the signals temporally in the latent space. Then, they begin to model each data set in the latent space as a mixture of some number of Gaussian components (or Bernoulli components, for the hand open/close statuses) of the same dimensionality as the data set. For parameters, the components each have means and covariances as well as a weight value that represents how likely it is that the data points were derived from that particular Gaussian. The algorithm attempts maximum-likelihood estimation of the mixture model's parameters using expectation maximization, but if the number of Gaussian components is unknown, Calinon et al. decide to estimate several different models with increasing numbers of components. They select one model to keep by testing the Bayesian information criteria (BIC) score (a measure of how well the model fits the data while minimizing the number of parameters involved) of each model. Specifically, Calinon et al. test models with up to ten components, keeping the model with the lowest BIC score.

From here, the algorithm by Calinon et al. (2007) begins to generate a general form for the input trajectories using Gaussian mixture regression. Essentially, time values are used as the query points for the mixture model to estimate the corresponding values in the latent space. Conditional expectations of the latent space values for a particular query point are calculated for each component using a combination of the mean for that component and elements from the covariance matrix for that component. Conditional covariances at that particular query point for each component are also calculated, this time using elements solely from the covariance matrix for that component. Calinon et al. explain that these conditional expectations and conditional covariances are mixed according to the probability that any particular Gaussian component contributes to the value for that query point. This probability is calculated for each component as a product of the weight of the component and the probability of the time value when used as a query point for the Gaussian described by the time-based parameters of that component (Billard, Calinon, Dillmann, & Schaal, 2008, p. 1378). This value, which is labeled as a Beta value, is then used as the weight for the component's conditional expected value in a weighted sum to determine the final general latent space value for the query point. The square of this Beta is used as the weight for the component's conditional covariance in a weighted sum to determine the final general latent space covariance matrix for the query point, as well (Calinon et al., 2007, p. 291). By incrementing the time values as query points, the algorithm generates a generalized motion for the learned trajectories with covariance matrices along this generalized motion; this is akin to the DTW-based naïve mean, but this time with covariances along the trajectory and a more robust *average trajectory* compared to the naïve mean.

While defining the latent space, Calinon et al. (2007) lay out a set of constraints for the trajectories learned by the robot, including constraints on the body of the robot and constraints in the environment (e.g. what the robot's initial arm configuration is and where the object's initial location is). After describing the Gaussian mixture regression step and retrieving the generalized form of the learned trajectory, they go on to find a trajectory that imitates the generalized motion while accounting for the body and environment constraints. To do this, they perform Lagrange optimization on a cost function for the candidate trajectories. The result is an iterative computation of the joint angles (in the latent space), after which they transform the joint angles back into the original data space and add back the means that were subtracted just before PCA.

5.3.2 Methods and Design

Calinon et al. (2007) demonstrate some empirical results showing successful learning of trajectories and variance data along those trajectories. We chose to form our system around their work because their results are very similar to our desired results and the algorithm happens to output a decent metric (covariances) for accuracy required along a learned trajectory. Again, I made use of Python to implement the algorithm's pipeline (see Fig. 4 at the end of this section for a diagram of the pipeline our Learning by Demonstration system uses). The Python function takes as input the set of trajectories that represent the task to be 'learned', again in the format outputted by the trajectory recording module. It also allows for specification of specific columns from the data, such that the learning algorithm can be run with just the joint angles, just the hand positions, or with any arbitrary data sets from the trajectory. Once the files are loaded in, the trajectory data is all concatenated into one large matrix. The time column is split away from this matrix, leaving only the data for the specified data set. I made use of the Scikit-learn PCA module (a Python module for performing PCA and other Machine Learning tasks), in which I specify that the number of components to extract should cover 98% of the original data spread, as in Calinon et al. (Pedregosa et al., 2011). After initializing the PCA object, I centered the data on zero (by subtracting the means of each dimension), then fitted and transformed the data to the latent space with the PCA object. I then concatenated the time column back to the data (the number of samples does not change). I split the data back up into separate trajectory recordings, this time in the latent space, and then I set the first trajectory loaded in as the reference signal and applied DTW to warp all the other trajectories to that reference (omitting the time column for this). Once this was complete, the time column from the reference signal was used as the time column for all signals (since they have been warped to the same time scale), and all the data was once again concatenated into a large matrix.

For initialization of the GMM, I once again made use of Scikit-learn, which has a GMM module. The function creates up to 10 Gaussian mixture models (with 1 through 10 components) and fits the large matrix of data to each model. For these GMM objects, I specified that the function should return the covariance matrices as full matrices, which is necessary for implementation of the Gaussian mixture regression. From each of these 10 models, I retrieved the weights, means, and covariances of each Gaussian component, as well as the BIC score, which exists as a method on the GMM object type in Python. Selecting the model (with *K* components) with the lowest BIC score for the data, I once again pulled out the time column from the reference signal. For the regression, I begin iterating through this time column, and for each time step, I calculated the expected value of the generalized point in the trajectory from the

K components and appended this point to the matrix in which I stored the generalized trajectory (the average trajectory). In addition, I calculated the expected value of the covariance matrix as well, and appended that to a list of covariances.

At this point, however, I performed an additional step that Calinon et al. (2007) do not do: I extracted the diagonal of the covariance matrix at each time step as a vector, and then took the square root of each element in that vector. The point of this operation is to extract a standard deviation from the generalized model: taking the diagonal of a covariance matrix retrieves the variances for each dimension under the assumption that dimensions are not correlated with each other, and taking the square root of the variance results in the standard deviation. This vector of standard deviations was appended to another matrix alongside the generalized trajectory, and the reason for using the standard deviation rather than the variances is because the inverse transformation for the PCA object can be applied to the standard deviation (since it is the same scale and dimensionality as the latent space data). The function then calls the PCA object's inverse transform method on both the generalized trajectory and the standard deviations (from the latent space to the original data space). As a last step, the function concatenates the time column back into both matrices and adds back the means that were subtracted from each dimension.

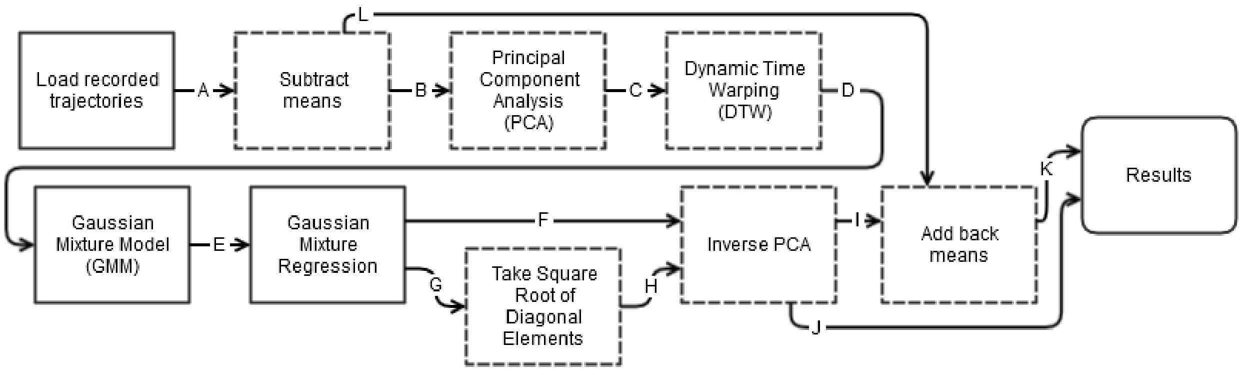
Load recorded trajectories
A
Subtract means
B
Principal Component Analysis (PCA)
C
Dynamic Time Warping (DTW)
D
L
Gaussian Mixture Model (GMM)
E
Gaussian Mixture Regression
F
G
Take Square Root of Diagonal Elements
H
Inverse PCA
I
J
Add back means
K
Results

Figure 4: Diagram of Learning by Demonstration system and data flow.
Components with dotted boxes indicate that the Time values are ignored during the calculation involved.
Data flow: A- Recorded trajectories, B- Zero-mean trajectories, C- Latent space trajectories,
D- Temporally-aligned latent space trajectories, E- Gaussian weights, centers, and covariances,
F- Generalized trajectory in latent space, G- Covariance matrices along generalized trajectory,
H- Standard deviations along generalized trajectory, I- Zero-mean generalized trajectory in original data space,
J- Standard deviations in original data space, K- Generalized trajectory in original data space,
L- Means of trajectory dimensions in original data space

5.3.3 Results and Discussion

The function returns several results, including the generalized trajectory and standard deviations in both the original and latent spaces, the covariances along the trajectory in the latent space, and the parameters of the Gaussian mixture model used for generation of the generalized trajectory. While Calinon et al. (2007) only return an ideal average trajectory with covariances, my extension of their algorithm, with the inclusion of the standard deviations, provides a metric of the accuracy needed during a task in the original data space (where the covariance is locked in the latent space). If the standard deviation at a particular point is low, then the trajectories that were used to train the generalized trajectory all seemed to pass through that point at that time (after DTW was applied). Furthermore, the standard deviations can be compared with the output of our accuracy parametrization task (with very little preprocessing), since the standard deviations are represented in the original data space. It is important to note that the Python implementation does not proceed past the Gaussian mixture regression step; where Calinon et al. continue on to retrieve joint angles for the learned trajectory that take into account the constraints they lay out, my implementation merely outputs the generalized trajectory and the standard deviations. This is acceptable for our project: we are more focused on learning the accuracy represented by the standard deviations rather than on learning the actual trajectory for a task.

Testing of this function revealed that it works as intended for learning trajectories and standard deviations along trajectories. Fig. 5 and 6 represent a set of trajectories in the latent space and original data space, respectively, with dotted lines as input trajectories and the solid line as the generalized trajectory (all dimensions are shown on the same axes). For these plots, note that error bars represent the width of one standard deviation for the standard deviations along the trajectory. In addition, note that Fig. 5 displays the input trajectories after PCA and DTW, while Fig. 6 displays the input trajectories without any processing steps such as DTW (hence why the standard deviations do not correlate as closely with the input trajectories). The plots reveal that as input trajectories converge, the standard deviation values along the generalized trajectory also drop.

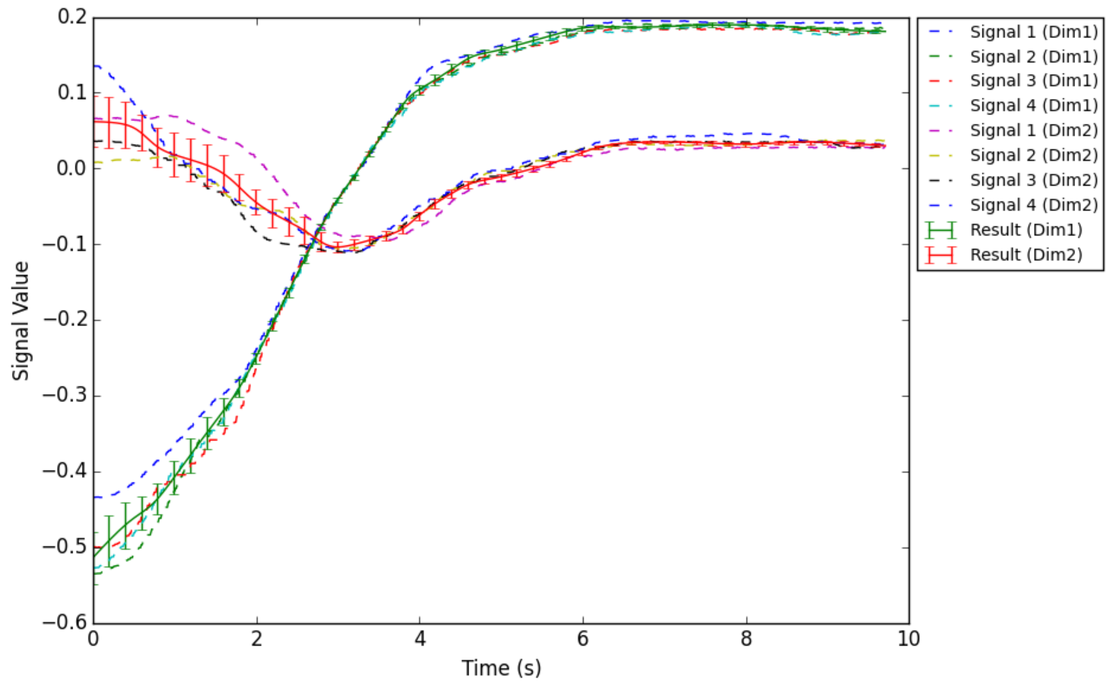

Figure 5: Learning algorithm output, latent space.
All signals except the Result signal have been processed by DTW to align with Signal 1.
Dim1, Dim2, etc. indicate the dimensions of the original data space.
Error bars indicate standard deviation values retrieved from algorithm,
and are displayed every 20 samples.

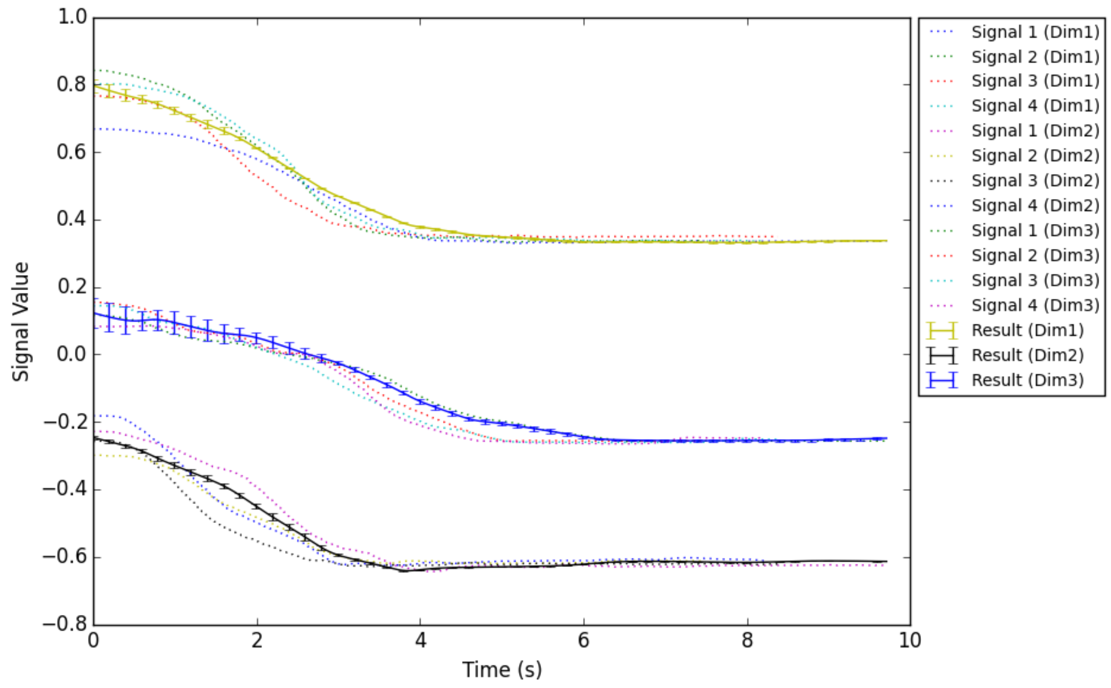

Figure 6: Learning algorithm output, original data space.
All signals except the result signal are displayed as they were recorded (no DTW).
Dim1, Dim2, etc. indicate the dimensions of the original data space.
Error bars indicate standard deviation values from the algorithm
transformed to the original data space, and are displayed every 20 samples.

The standard deviation results provide us with a clear method for determining the accuracy required to perform a task at various points along the task trajectory. This is a completion of a fundamental part of our system, as the accuracy-inferring behavior relies on the data output from this function. Further work on the algorithm may include generalizing the implementation to allow for environmental constraints and new contexts by implementing the rest of the steps outlined by Calinon et al. (2007). However, our learning algorithm is sufficiently complete for our project's goals, so our next steps include implementation of a real-time accuracy comparison on the Baxter robot, where Baxter executes a generalized trajectory for a task and at each time step compares the accuracy bounds reflected by its current joint angles to those bounds specified by the standard deviations. Once that is complete, we will have achieved our goal of a software system that is able to learn a task's accuracy requirements and take steps to account for those accuracy requirements.

6 CONCLUDING REFLECTIONS

(Author: Srinivasan)

Our capstone project's initial development pace was slow, mainly due to the learning curve involved with developing software for the Baxter robot and the literature review necessary to begin implementing algorithms, but after familiarizing ourselves with the software and hardware components, our efficiency increased dramatically. When planning project management goals for each semester, we had to revise our initial plans to consider this curve, but because we initially planned for a very flexible development system, where features were developed independently and iteratively, we managed to stay on track. Insights gained from this process included the importance of division of the project goal into accomplishable tasks: we were able to complete the parts of our system solely because the engineering tasks were divided into parts that could proceed independently. We also learned that having multiple people responsible for the completion of a single task resulted in less efficient work, as meeting and collaboration overhead could stifle actual development work. We rectified our task division later in the development process to account for this, such that each person could work on a task separately. By no means did this prevent collaboration, but it did allow us to complete tasks independently, which, as stated before, was crucial to our project's success.

Future work for our system can include generalization to other robotic hardware; we attempted to keep the data pipeline as universal as possible, but some components still rely on Baxter's features. Such improvements would bring the system closer to the market-viable product we laid out earlier. Meanwhile, future research in the same area would benefit from the setup we have already laid out, where trajectories can be recorded and processed easily, then analyzed according to the metrics necessary in such research. Those metrics can then be compared using the hardware's known parameters and our accuracy parametrization system, providing a result similar to our current results, but for whichever metrics the research wishes to compare.

Wrapping up, the current outcomes that our team has for this capstone project, for the most part, align with the deliverables of our original plans. We set out to generate a system that could generalize a set of robot behaviors into a single action that contained data on the accuracy required at certain points during that action, as well as a way to compare that generalized action to our knowledge of the robot's abilities. Our result, which included a generalized trajectory with standard deviation data along the trajectory as well as tolerance information for any configuration of the Baxter robot's joints, fits that description perfectly.

References

(Strategy References:)

Bi, F. & Mac, R. (2015). Drone Maker 3D Robotics Raises $50 Million in Latest Round. *Forbes*. Retrieved from http://www.forbes.com/sites/frankbi/2015/02/26/drone-maker-3d-robotics-raises-50-million-in-latest-round/

Biesada, A. (n.d.). Hoover's Company Profiles: iRobot Corporation. *Hoover's, Inc.* Retrieved from http://subscriber.hoovers.com/H/company360/fulldescription.html?companyId=132607000000000

Brault, M. (2012). Americans With Disabilities: 2010 - Household Economic Studies. *Economics and Statistics Administration, US Department of Commerce*. Retrieved from http://www.census.gov/prod/2012pubs/p70-131.pdf

Censky, A. (2011). Older Americans are 47 times richer than young. *CNN Money* - *The New American Dream*. Retrieved from http://money.cnn.com/2011/11/07/news/economy/wealth_gap_age/

Crompton, J. (2014). IBISWorld Industry Report 33399: Power Tools & Other General Purpose Machinery Manufacturing in the US. *IBISWorld.* Retrieved from http://www.ibis.com

Ellis, B. (2013). Nursing home costs top $80,000 a year. *CNNMoney (New York)*. Retrieved from http://money.cnn.com/2013/04/09/retirement/nursing-home-costs/

Hoover's, Inc. (n.d.). Hoover's Company Profiles: Clearpath Robotics Inc. *Hoover's, Inc.* Retrieved from http://subscriber.hoovers.com/H/company360/overview.html?companyId=244260399

Hoover's, Inc. (n.d.). Industry Report: Home Health Care Services. *Hoover's, Inc*. Retrieved from http://subscriber.hoovers.com/H/industry360/overview.html?industryId=1383

McMillan, R. (2015). Now We Can Build Autonomous Killing Machines. *Wired*. Retrieved from http://www.wired.com/2015/02/can-now-build-autonomous-killing-machines-thats-bad-idea/

Owano, N. (2011). Willow Garage slashes price (and arm) of PR2 robot for research. *PhysOrg*. Retrieved from http://phys.org/news/2011-08-willow-garage-slashes-price-arm.html

O’Donnell, F. (2015). Issues and Insights. *Mintel: Senior Lifestyles - US - December 2013*. Retrieved from http://academic.mintel.com/display/689700/

Ortman, J. M., Velkoff, V. A., & Hogan, H. (2014). An Aging Nation: The Older Population in the United States. *Economics and Statistics Administration, US Department of Commerce*. Retrieved from http://www.census.gov/prod/2014pubs/p25-1140.pdf

Porter, M. E. (2008). The Five Competitive Forces That Shape Strategy. *Harvard Business Review 86*(1), 25-40.

Rethink Robotics (2015). Build a Baxter. *Rethink Robotics.* Retrieved from http://www.rethinkrobotics.com/build-a-bot/baxter/

Tobe, F. (2013). A glimpse at our robotic future: 235 start-ups reviewed. *Robohub*. Retrieved from http://robohub.org/a-glimpse-at-our-robotic-future-235-start-ups-reviewed/

(IP References:)

Barajas, L. G., Martinson, E., Payton, D. W., & Uhlenbrock, R. M. (2014). Method and system for training a robot using human-assisted task demonstration. *U.S. Patent No. 8,843,236*. Retrieved from https://www.google.com/patents/US8843236

Billard, A., Calinon, S., Dillmann, R., & Schaal, S. (2008). Robot Programming by Demonstration. *Springer Handbook of Robotics*, 1371-1394.

Brown & Michaels, PC. (2006). How do I read a patent? - the Claims. *Brown & Michaels*. Retrieved from http://www.bpmlegal.com/howtopat5.html

Klein, A. R. (2015). Intellectual Property Basics for Technology Companies. *Latham & Watkins*.

Ravindran, B., Barto, A. G., & Mathew, V. (2007, January). Deictic Option Schemas. In *IJCAI* (pp. 1023-1028).

(Technical Contributions References:)

Abbeel, P. & Ng, A.Y. (2002). Apprenticeship Learning via Inverse Reinforcement Learning. *International Conference on Machine Learning, 21*. Retrieved from http://ai.stanford.edu/~pabbeel/pubs/AbbeelNg_alvirl_ICML2004.pdf.

Billard, A., Calinon, S., Dillmann, R., & Schaal, S. (2008). Robot Programming by Demonstration. *Springer Handbook of Robotics*, 1371-1394.

Calinon, S., Guenter, F., & Billard, A. (2007). On Learning, Representing, and Generalizing a Task in a Humanoid Robot. *IEEE Transactions on Systems, Man, and Cybernetics, 37*(2), 286-298.

Müller, M. (2007). Dynamic Time Warping. *Information Retrieval for Music and Motion*, 69-84. doi:10.1007/978-3-540-74048-3_4.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research, 12*, 2825-2830. Retrieved from http://scikit-learn.org.

Rethink Robotics. (2015). Build a Baxter. *Rethink Robotics*. Retrieved from http://www.rethinkrobotics.com/build-a-bot/baxter/.

Schweigert, S. (2015). Trajectory Recording and Playback. Unpublished manuscript, Department of EECS, University of California, Berkeley.