Journal of Robotics and Autonomous Systems
v. 22, 1997, pp. 361-375

TOWARD GESTURE-BASED PROGRAMMING: SHAPE FROM MOTION PRIMORDIAL LEARNING OF SENSORIMOTOR PRIMITIVES

Richard M. Voyles
Computer Science Department
University of Minnesota
Minneapolis, Minnesota

J. Dan Morrow
Pradeep K. Khosla
Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania

ABSTRACT

Gesture-Based Programming is a paradigm for the evolutionary programming of dextrous robotic systems by human demonstration. We call the paradigm "gesture-based" because we try to capture, in real-time, the *intention* behind the demonstrator's fleeting, context-dependent hand motions, contact conditions, finger poses, and even cryptic utterances, rather than just recording and replaying movement.The paradigm depends on a pre-existing knowledge base of capabilities, collectively called "encapsulated expertise," that comprise the real-time sensorimotor primitives from which the run-time executable is constructed as well as providing the basis for interpreting the teacher's actions during programming. In this paper we first describe the Gesture-Based Programming environment, which is not fully implemented as of this writing. We then present a technique based on principal components analysis, augmentable with model-based information, for learning and recognizing sensorimotor primitives. This paper describes simple applications of the technique to a small mobile robot and a PUMA manipulator. The mobile robot learned to escape from jams while the manipulator learned guarded moves and rotational accommodation that are composable to allow flat plate mating operations. While these initial applications are simple, they demonstrate the ability to extract primitives from demonstration, recognize the learned primitives in subsequent demonstrations, and combine and transform primitives to create different capabilities, which are all critical to the Gesture-Based Programming paradigm.

1. INTRODUCTION

Once implemented, many robotic applications fail to meet expectations or prove insufficiently robust for the desired level of autonomy. A significant reason for this is that programming difficulties maintain a layer of insulation between the system's use and it's development. Because the programming requires so much expertise, unskilled users, by definition, can not be programmers and experienced programmers are too "valuable" to be regular users. As a result, there is a constant gap between personnel needs and expertise, both on the part of the programmer and user.

Numerous attempts have been made to ease the discomfort of robot programming with varying degrees of success. Behavior-based and multi-agent systems [2] seek to modularize software for easier programming by programming experts. Visual programming environments like Chimera/Onika [28][4] and commercial packages such as MatrixX/SystemBuild and LabView capitalize on modular, reconfigurable software blocks by iconifying them within "software assembly" environments. These visual environments allow programming at a much higher level, hiding many details of the particular implementation, and lessening the burden of programming expertise.

Despite these advances, a fairly high level of expertise is still required to program and interact with robots. For example, Onika allows novice users to easily build simple pick-and-place applications in a matter of minutes [3]. But, it is imperative that the user understands and consciously considers the importance of via points, collision avoidance, grip selection, and the dynamic effects of transport in order to for the application to be successful. Since these characteristics are inherent in all manipulative tasks and are adeptly mastered by humans, they are usually handled unconsciously. Hence, the distinction between task experts and programming experts. Programming experts are trained to bring these subconscious requirements up to the conscious level in order to transcribe them as required by the particular programming environment.

Humans are more effective at teaching by demonstration and practice [19]. Therefore, recent approaches to robot programming have been aimed at *learning by observation* [9],[13]. By forcing the robot to observe a human interacting with the world, rather than forcing the human to interact with a *textual representation* of the robot interacting with the world, a more natural, "anthropocentric" environment results. Essentially, this is an extension of the popular "lead-through teaching" [29] and "point-teaching" approaches used in industry. These approaches are used to program spray painting and welding operations, respectively, because they are easily mastered by semi-skilled workers. Lead-through teaching, in particular, allows anyone who can perform a valid task (however "valid" is defined by the application) to program the robot to perform the task by demonstrating

it. As such, programming changes can be made reactively or proactively by a task expert, not by calling in a robot programming expert.

2. GESTURE-BASED PROGRAMMING OVERVIEW

Gesture-Based Programming (GBP) is a form of programming by human demonstration that extends the work of Kang and Ikeuchi on "Robotic Instruction by Human Demonstration" [9]. The human (stick figure on left of Figure 1) demonstrates the task while wearing a sensorized glove. The modular glove system senses hand pose, finger joint angles, and fingertip contact conditions. Objects in the environment may or may not be sensed with computer vision while a speech recognition system extracts "articulatory gestures." (Gestures will be described later but constitute all actions of the demonstrator.) Primitive gesture classes are extracted from the raw sensor information and passed on to a gesture interpretation network. These autonomous multi-agent networks extract the demonstrator's intentions with respect to the system's skill base to create an abstraction of the task. In other words, the system is not merely remembering everything the human does, but is trying to understand -- within its scope of expertise -- the subtasks the human is performing ("gesturing"). These primitive capabilities in the skill base take the form of *encapsulated expertise agents* -- semi-autonomous agents that encode sensorimotor dexterity.

The output of the GBP system is an executable program for performing the demonstrated task on the target hardware. This program consists of a network of encapsulated expertise agents of two flavors. The primary agents implement the primitives required to perform the task and come from the pool of primitives represented in the skill base. The secondary set of agents includes many of the same gesture recognition and interpretation agents used during the demonstration. These agents perform on-line observation of the human to allow supervised practicing of the task, if desired. (Stick figure on far right of Figure 1.)

As mentioned above, the human model for teaching by demonstration most often involves a practice phase. The reason for this is that passive observation of a task rarely provides accurate parametrization for the sensorimotor primitives of the trainee's deduced task "model." Incorporating gesture recognition and interpretation agents into the executable provides an intuitive way for the demonstrator, or another user, to fine tune the operation of the program without having to demonstrate the entire task over again.

What are gestures?

Gestures, within the scope of this work, are defined as "imprecise events that convey the intentions of the gesturer." There are many gestural modalities including *symbolic gestures*, such as the pre-shape of a dextrous grasp, *motion gestures*, such as trans-

porting or manipulating an object, *tactile gestures*, such as fingertip contact force, and *articulatory gestures*, such as “oops!’ when an error is made or geometric model-based information such as “tighten the *screw*.” (The whole phrase is irrelevant; only the keyword “screw” is important.) Other gestural modes exist but this is the subset we consider for our GBP system. Because gestures are imprecise events, they are context dependent; in isolation they convey little information. Yet, when combined with the state of the system and the environment at the time the gesture occurred, perhaps augmented with some past history of gestures, the intention becomes interpretable.

In effect, gestures form an “alphabet” from which “words” are formed when combined with state information. The state information is also represented by characters in the gestural alphabet. These gestural words are strung together by the gesturer into “sentences” which form the basis for interpretation. The gestural alphabet is like a linguistic alphabet in the sense that characters are parametrized. For instance, in English, the letter “a” has many sounds including the long “a” as in “save,” the short “a” as in “sad,” and the silent “a” as in “read.” This is an implicit parametrization. In contrast, the gestural characters are explicitly parametrized by either a single integer or floating-point argument.

3. WHERE DOES ENCAPSULATED EXPERTISE COME FROM?

The focus of GBP is on tasks involving substantial environmental contact. A fundamental assumption on which the paradigm is based is the availability of a knowledge base of *a priori* sensorimotor dexterity -- pre-compiled capabilities the target system already “knows.” The premise is that programming is a form of teaching and teaching by demonstration is much more natural and effective for human teachers. But, teaching is virtually impossible without some assumption of *a priori* capabilities on the part of the student. For example, in teaching calculus, one assumes competency in algebra and the demonstration of the task of, say, taking derivatives, is performed as a sequence of these basic algebraic primitives. The student observes the demonstration and interprets it in terms of these primitive capabilities. Hence, the need for the skill-base at all. It provides the “shared ontology” that is the basis of interpretation and communication between teacher and student.

For GBP, the origin of this “skill base” is not particularly important. It is only important that the primitives exist and that they can be related to observations of the demonstration. An additional desirable property is that they can be related to each other. This implies the existence of a similarity measure and a mechanism by which primitives can be combined or transformed to produce new primitives. As we will see, Shape from Motion Primordial Learning provides these capabilities.

There are three basic mechanisms by which these primitives can come into existence. The first is “clairvoyance.” The user (or automatic planner), acting as a god, hand codes a primitive that he/she *knows* is useful and composable. An example of this is a guarded move. The second mechanism is through assembly of lower-level primitives. Using the calculus analogy, algebraic primitives may form the basis for calculus, but arithmetic primitives form the basis for algebra. Finally, there is learning.

In this paper, we present a learning approach for extracting sensorimotor primitives from teleoperated manipulations that is a natural extension of an autonomous sensor calibration technique called *Shape from Motion* [37]. Calibration is, after all, a form of learning; one wants to “learn” the transformation from the sensor’s input space to the output space. Most traditional calibration techniques (such as least-squares) do not pose it as a learning problem because it is overly constrained: apply a known force, take a measurement; apply a known force, take a measurement. Shape from Motion calibration extracts the “shape “of the input/output mapping that results from random “motion” of the input through the excitation space. The result is calibration of the sensor with little or no knowledge of the loads applied to the sensor. It became apparent that a derivative of shape from motion could be applied to other types of learning problems.

Shape from Motion is based on the extraction of an eigenspace representation of the input/output mapping. It uses singular value decomposition to derive principle components. In the case of calibration (described in more detail in Section 7), the principle components analysis (PCA) is constraint-directed, so model-based information can be incorporated into the learning process. Because the underlying representation (eigenvectors) is linear, it is limited to learning linear, or at least piece-wise linear, interactions. This is a drawback because it complicates the segmentation of the observations into linear, learnable subtasks. The advantage is the representation provides not only a mechanism to learn the primitives, but natural mechanisms for identifying the primitives in subsequent observations and combining and transforming existing primitives into new primitives. These are abilities many other learning approaches have difficulty demonstrating.

4. PRIOR WORK ON LEARNING PRIMITIVES

We are interested in developing a primitive representation which supports acquisition, integration, and transformation of sensorimotor primitives. Previous work in learning robot skills is relevant to this effort because it provides potential primitive acquisition techniques and because primitives learned by different techniques than primordial learning may also be used in the execution phase of our system. The idea is to relate primitives to recurring subgoals of complex tasks and to reuse primitives in the construction of complex task strategies, whatever the origin of the primitives may be.

Speeter [26] developed kinematic primitives for the control of the Utah/MIT Dextrous Hand. These primitives encapsulated useful, coordinated finger trajectories which were used to construct more complex task strategies. Michelman and Allen [17] also developed complex task strategies from task primitives. Paetsch and von Wichert [20] have developed a set of behaviors which can be combined to perform a peg insertion with a multi-fingered hand. These primitives focused more on kinematics and the coordination of many degrees of freedom. Our focus is more on relating sensor feedback signals to the motor commands.

Yang et al [38] have applied hidden markov models (HMM) to skill modeling and learning from telerobotics. The skills learned are manipulator positioning tasks without force or vision feedback. Some researchers are applying supervised learning approaches to recover strategies for contact tasks like deburring. Liu and Asada [15] use a neural network to recover an associative mapping representing the human skill in performing a deburring task. The task is performed using a direct-drive robot with low friction and a force sensor integrated with the workpiece. This allows the human to perform the task during the training phase will little interference from the robot while much relevant information is measured by the robot sensors. Shimokura and Liu [24] extend this approach with burr measurement information.

Along these lines of skill acquisition by supervised learning is ALVINN[21], an artificial neural network for vehicle steering. ALVINN has demonstrated robust mastery over a wide range of vehicles and road types by observing a human drive the vehicle on the target road type. For the purpose of comparison, Hancock and Thorpe developed ELVIS [6], a PCA-based vehicle steerer to operate on the same vehicles. The success of ELVIS provided additional impetus to continue this work.

Several researchers have applied reinforcement learning methods to the recovery of a peg insertion skill. Simons et al [25] learn how to interpret a force feedback vector to generate corrective actions for a peg insertion which has an aligned insertion axis. The output motion commands are restricted to the plane normal to the insertion axis. Changes in force are used to reinforce (penalize) the situation/action pair. Gullapalli et al [5] learned close tolerance peg insertion using a neural network and a critic function consisting of the task error plus a penalty term for excessive force. The input to the network was the position vector and force vector and the output was a new position vector. About 400 training trials are required to recover a good strategy. However, the learned “skill” is specific to the peg geometry which it is trained on and is specific to the location of the peg in the workspace because the absolute peg position is produced as output. If the peg location were moved in the workspace, this skill would probably fail because it would be very difficult to accurately (relative to the insertion clearance) specify the relative trans-

formation between the new location and training location. A few more training trials would probably suffice for learning the new skill, but one does not want to learn and store a different skill for every location in the robot's workspace.

Vaaler and Seering [32] have applied reinforcement learning to recover production rules (condition-action pairs) for performing a peg insertion task. The critic function is a measure of the forces produced from the last move increment; higher forces are penalized. The termination conditions are an absolute Z position (Z is the insertion axis) and a Z force large enough not to be caused by 1 or 2 point contact (common during insertion). Ahn et al [39] learn to associate pre-defined corrective actions with particular sensor readings during iterative training and store these mappings in a binary database. Again, the critic function penalizes moves which increase the measured force. Kinematic analysis of the task can be used to "seed" the database with a priori knowledge, but this is not necessary for the method to succeed. Lee and Kim [14] propose a learning expert system for the recovery of fine motion skills. Skills are represented as sets of production rules and expert a priori knowledge is used. The critic function is the distance between the current state and the goal state, but does not explicitly include an excessive force penalty. The method is tested on a simulated 2D peg insertion task.

5. PRIMORDIAL LEARNING

Primordial Learning refers more to a way of thinking than to a specific algorithm; it refers to learning fundamental interactions, or mappings, with no prior knowledge. It's like an infant that learns gross motor control by flailing. The infant is aware of "sensor and actuator ports," but has no comprehension of their connection to the body or to the outside world. Some might say primordial learning is non-parametric learning. Some might call it unsupervised learning. The implementations in this paper can be called principle components analysis. Yet, all these terms, broad and narrow, are, in one way or another, inaccurate in describing our work in its entirety, from autonomous sensor calibration to mobile robot behaviors, hence, the different label.

Mobile Robot Primitives

The infant analogy provided not just the impetus for the name, but the impetus for the first application. Somewhat as a lark, the challenge was made to create a mobile robot that could learn to "crawl" before the newborn infant of the first author could do so. The term "crawl" was interpreted loosely to mean "purposefully motivate" so as not to exclude the creation of a wheeled vehicle as opposed to a much more mechanically complex legged vehicle.

This sort of primordial robot learning has been suggested for mobile robots before. Pierce [22] taught a simulated robot to navigate around obstacles and even to home in on a goal without any explicit knowledge of what the sensor and actuator data

meant. Likewise, Maes and Brooks [16] developed a subsumption network that allowed a physical legged robot to learn to walk. Of course, both of these examples employed a specific objective function that guided the learning process to the desired outcome.

In our implementation, we applied the shape from motion primordial learning technique to a small mobile robot that has no explicit knowledge of its limited set of actuators and sensors. All the robot is "aware of" is the streams of data that come from or go to the sensors and actuators. The shape from motion technique allows the robot to develop an internal representation of teleoperated interaction between sensors and actuators to produce "meaningful" externalized behavior. In this sense, the robot is primordial, or infant-like, because it must learn the most basic input/output relationships from a teacher, fuse them as required to mimic the behavior, and ignore superfluous data. We do not use an explicit objective function or provide a reinforcement signal, but the learned result is explicitly dependent on the teleoperated interactions presented by the teacher.

The shape from motion technique as applied to this problem uses the standard principal component analyses of Hancock and Thorpe [6] and Pierce [22]. The standard principal components analysis does not include a model-based constraint as will be investigated later. For primordial learning, we do not wish to constrain the problem. Although some constraints can be useful to achieve more sophisticated results (as in [22]), they impose potential limitations which may not be expressive enough for the situation at hand.

The subject of this experiment is the MK-V, a three-wheeled, non-holonomic mobile robot with a single drive wheel and an unpowered steering wheel. Sensors include wheel encoders, a compass, bump sensors and motor current sensors. The drive and steering motors have three-valued -- forward-off-reverse -- commands with no closed-loop controllers on velocity or position. For this initial experiment there are no redundant groups of very similar and correlated sense elements such as a visual retina (as in [6]) or sonar ring (as in [22]).

With no algorithmic structure imposed at all on the fusion process, the result learned is dependent on what the robot is allowed to observe. For training, we teleoperated the robot in a cluttered environment allowing it to wander while bumping into obstacles and jamming its wheels.

The training data consists of a matrix of input/output vectors sampled periodically during teleoperation. The input/output vector is composed of the actuator commands concatenated to the sensor data. The mean of this vector, **a**, over the entire sequence was subtracted out to normalize the data values. Next, the matrix was batch-processed using SVD to extract the eigenvectors

and the largest *n* eigenvectors were selected using the largest ratio of adjacent singular values as the threshold for *n*. This test assumes there is a group of "significant" eigenvectors, $\mathbf{e}_i$, with similar singular values and a group of "insignificant" eigenvectors with small singular values that represent the noise. Examining the ratios of adjacent, ordered singular values indicates the dividing line between the groups. For practical reasons, we also draw a line between groups if any singular value is more than 20 times smaller than the next largest. This is a heuristic derived from the sensor noise and quantization of the A/D's. These two criteria together give us an absolute and a relative measure of significance of the eigenvectors. For run-time operation, the new sensor image, **x**, is projected onto the eigenspace according to the following equation as described in [6]:

$$\mathbf{v} = \mathbf{a} + \sum_{i=1}^{n} ((\mathbf{x} - sensor(\mathbf{a})) \bullet sensor(\mathbf{e}_i))\mathbf{e}_i \qquad (1)$$

where "sensor(**e**)" refers to only the sensor elements (as opposed to the actuator elements) of the vector **e**.

Using this technique, the real robot successfully achieved both wandering behavior and escaped stalls and collisions without highly redundant or correlated sensors and with no algorithmic structure imposed on the learned result. Figure 2 shows a teleoperated run of the MK-V with comparisons of what the human did and what the MK-V would have done on its own with eigenvectors learned on a prior training set. The drive motor commands of the Shape from Motion technique closely match the commands of the human. Furthermore, the steering motor learned *when* to steer, but the steering *direction*, which is random and superfluous to getting un-stuck, does not match. This fused result is appropriate; superfluous information and, in fact, entire superfluous sensors were ignored to achieve the proper trained behavior.

We are currently integrating a line-scan camera onto the MK-V to see if it can learn line tracking for a CMU "Mobot" competition. The line-scan camera is fundamentally different because it provides a dense array of identical sensing elements. We are confident it will be successful because of its similarity to the visual retina and sonar array of [6] and [22], respectively.

PUMA Manipulator Primitives

While the application on the MK-V mobile robot started out as a lark, learning sensorimotor primitives for robot manipulation skills is more germane to the body of research conducted in our lab. Automatic generation of usable primitives is a necessary technology demonstration for the credibility of Gesture-Based Programming. Because the skill base can incorporate primitives from a variety of sources, it is not necessary to generate all primitives automatically. Nonetheless, it is important to show proof-of-concept of the end-to-end system.

Guarded Move. As a first demonstration, we attempted to learn a one-dimensional guarded move. Although this is a fairly trivial primitive that can easily be hand-coded, it is not quite as obvious how to robustly identify it during human demonstration. This is the strength of an integrated approach like shape from motion primordial learning; it provides primitive identification and transformation as well as the basic learning of the primitive.

To learn **zguard** -- a guarded move along the z-axis -- we teleoperated a PUMA robot with a 6-DoF force/torque sensor (calibrated with the shape from motion paradigm, incidentally [37]) so that it came into contact with a table while moving along the approach (z) axis of the end effector. We repeated this one-dimensional guarded move ten times, logging data only during the guarded move, not during the retraction phase when the end effector was moved away from the surface. Average approach height was about 40 mm for this first training dataset. The robot was controlled by a cartesian velocity controller, teleoperation input was provided by a 6-DoF trackball, and operator feedback was visual as well as a graphical display of the real-time force/torque components.

The input/output vector from which the data matrix for training was generated consisted of the useful data -- 6 measured force components, the total force and torque magnitudes, and 6 commanded cartesian velocities -- plus some irrelevant data -- 3 cartesian position elements and 9 cartesian orientation elements. We used the same algorithm for extraction of the eigenvectors as used on the MK-V. The singular values that resulted from the training data are plotted in Figure 3.

The output of the training was a primitive of one eigenvector that performed very well. Figure 4 shows the measured force components as the guarded move primitive autonomously acquires a hard surface. 20 N was the target force threshold applied during training and this can be varied by scaling the output portion of the extracted eigenvector. The primitive worked equivalently in all trials performed regardless of region of workspace, orientation of the end effector, compliance of the surface, or external perturbations. Analysis of the components of the eigenvector support this. The primitive is looking at the *z*-component of force as well as the total magnitude of the force.

To identify instances of the primitive from demonstration data, we run the training algorithm on windowed batches of runtime data and examine the parallelism between the extracted eigenspace and the eigenspace representation of the primitive. The dot product gives a quantified measure of eigenvector parallelism that we compare to a high threshold for presence or absence of the primitive.

To test the recognizer, we randomly moved the robot around the workspace, occasionally coming into contact with surfaces. The commanded velocities appear in Figure 5 while the corresponding force measurements are plotted in Figure 6. The resulting goodness-of-fit is plotted in Figure 7.

As evidenced by the figures, the goodness-of-fit measure performed just as it should, picking out every instance of a guarded move in z and rejecting every instance of random motion or random contact, including guarded moves along other axes. The occasional dropouts to zero are inconsequential as this fitness measure is but one input to the gesture interpretation network of the Gesture-Based Programming system.

For gesture-based programming we want to be able to identify primitives from natural hand motions rather than teleoperations using a trackball. Although learning primitives can be done either way, identification must be in a more natural environment. To do this, we must instrument the human's hand with force and velocity sensors as described in Section 2. Since we are still fabricating wearable force sensors [33], we used a standard 6-axis force sensor held in the human's hand. Differentiating and filtering the output of a Polhemus device from the CyberGlove provided cartesian position and velocity.

The recognition results for the guarded move are shown in Figure 12. The bold line indicates the confidence of match between the human motions and the previously learned **zguard** primitive. As above, it is the parallelism between primitive eigenvectors and the principal components of the new data. For reference, z-force and x- and y-force are plotted on the same axes (scaled down by a factor of 25). The matcher correctly picked out all three instances of **zguard** and correctly rejected all instances of **yguard** and **xguard** as well as all retractions from the table.

Edge-to-Surface Alignment. The guarded move is a "move until force threshold" operation. Movement is an explicit part of the goal. Alignment operations, on the other hand, only move in order to accommodate. Since no explicit motion is part of the primitive, it is difficult to teleoperate. What we'd like to do is use the previously learned guarded move to bring the edge and surface together while we teleoperate only the alignment primitive. The problem is the learned result would include both primitive operations, since there is no way to separate the sensor stream into "guarded move" and "alignment" portions. However, the eigenspace representation provides a convenient mechanism for separating the two after learning, if they are truly elemental and decoupled.

Learning the "yroll" alignment task -- alignment of an edge to a surface by accommodating rotationally around the y-axis -- was accomplished in the same manner as the "zguard" primitive with the exception that zguard was running during the teleop-

eration phase. This complicated the eigenvector extraction, as well. We expect the zguard eigenvector to appear in the new set of trained vectors and, in fact, it does. Using our standard ratio of eigenvalues test, only one eigenvector is extracted during training and its dot product with zguard is 0.9993, indicating it is the **zguard** primitive.

To extract the alignment primitive, we must look to the next significant group of eigenvectors. Ignoring the **zguard** eigenvector produces a set of four eigenvectors that very nicely implement the desired accommodation operation. Figure 8 shows the commanded velocities for an experimental run with both the zguard and yroll primitives superposed. The corresponding force plots (Figure 9) show the desired result of moving in z to press the edge flat against the table was achieved. Simple superposition of eigenvectors allows for decomposition and recomposition of the primitives at least in this case of decoupled operations.

Surface-to-Surface Alignment. The above decoupled primitives can be combined to produce more complex behavior. Combining **zguard**, **xroll**, and **yroll** produces a "skill" that allows two surfaces to be pressed together. An experimental run of these three primitives is shown doing just that in Figure 10 and Figure 11.

6. SHAPE FROM MOTION CALIBRATION

Calibration is a form of learning, as previously asserted; it is just a little more structured than most learning problems. Traditional least squares calibration is not posed as a learning problem, however. Least squares calibration requires the user to apply a known set of loads and to measure the corresponding outputs. The data is merely reduced to find the best gains that maximally agree with the pair-wise matchings of input and output.

We have demonstrated that applying shape from motion primordial learning to the problem virtually eliminates the requirement of knowing all the applied loads [37]. Incorporating a geometric, model-based constraint into the eigenspace learning technique allows the "shape" of the force sensor (the input/output mapping) to be extracted only from the raw sensor measurements while a random, unknown load vector of constant magnitude is moved through the input space. Hence, the *shape* is learned from *motion*.

A flow chart of the calibration procedure using shape from motion primordial learning appears in Figure 13. Data is gathered and eigenvectors are extracted by SVD. Model-based information -- the dimensionality of the force sensing space -- is employed to select the correct number of significant eigenvectors. Then the motion constraint, a geometric model based constraint, is employed to transform the eigenvectors into the desired sensing space. The left and right eigenvectors result in the shape and mo-

tion matrices. The final step employs a few know applied loads to scale the result to the desired engineering units and orient it with respect to the desired reference frame.

EXPERIMENTAL COMPARISON TO LEAST SQUARES

As reported in [37], we applied our technique to two different sensors and compared the results to the standard least squares approach. We wanted to demonstrate that even though it is posed as a learning problem, the results are just as accurate, if not more so, than the highly structured approach and take less time to acquire. The first sensor was a 4-DoF fingertip force sensor that we could easily isolate to look only at pure forces in the plane. The other was a 6-DoF commercial wrist force/torque sensor.

Fingertip Results

For the shape from motion approach, we chose a mass in the linear region of the sensor, near the upper extreme of what we consider "reasonable."[1] We picked the upper extreme to improve signal-to-noise, but we stayed within the "reasonable range" to reduce any effects of minor nonlinearities. This is the extent to which we considered experiment design because we assumed the sensor is highly linear. Subsequent testing showed this assumption is valid. The chosen test mass weighed in at 111.4 grams and the calibration procedure took only 1.5 minutes while collecting a total of 96 unknown data points and 1 known load.

For the least squares approach, we chose five different masses, each of which was applied at 40-degree increments around the circle for a total of 45 accurate loads. For each load we gathered several measurements and averaged them to reduce the effects of measurement noise. We chose a large number of loads to reduce the effects of applied load error. The same criteria were used in selecting the calibration masses which were: 60.4, 78.3, 95.2, 111.4, and 122.8 grams. The procedure consisted of adjusting the angle of the rotating stage to within 0.2 degrees, sequentially applying the five masses, and then incrementing the angle. The entire procedure took 17 minutes to collect all 45 known data points (including redundancy at each data point).

Because of the nearly exhaustive nature of the least squares calibration, the time difference is more than 10:1. But even cutting the procedure down to 15 measurements results in nearly a 4:1 time advantage for shape from motion, not to mention the loss in accuracy for least squares due to lesser noise rejection. The large number of least squares measurements gathered should produce nearly the best possible least squares calibration.

To assess the calibration results quantitatively, we hung each of the five masses used for the least squares calibration on the string and gathered raw data as the sensor was rotated 360 degrees. We then calculated the force magnitude at each sample point

1. Reasonable in relation to the tasks we intend to tackle.

and performed two calculations. The first determined the average magnitude to assess absolute accuracy. The second smoothed the magnitude signal to eliminate high frequency noise and determined the standard deviation of the smoothed signal. This quantifies the “imprecision” of the measurement across all orientations for a constant magnitude force. The best and worst of these data are tabulated in Table 1.

Lord F/T Results

For the 6-DOF sensor we built a special calibration fixture to help us make quick, accurate measurements. It consists of a 3-inch aluminum cube (approximately 1 kg) and two 1-inch diameter brass bars (approximately 1 kg each). The cube has one threaded hole in the center of each face into which the bars can screw. This provides 16 unique force/torque combinations that can be quickly and precisely selected during calibration. The simple design is easy to machine yet it dramatically reduces the time required for the least squares approach because the flat faces provide an accurate reference surface for leveling the device in various orientations. It is quite convenient for the shape from motion approach as well.

Despite using the calibration fixture and collecting only 13 load vectors (force magnitudes of 12.09N, 22.9N, and 33.71N with different moments), the least squares approach required 69 minutes of data collection time. (Without the calibration fixture, using our own version of the “torque bar” as in [31], took even longer.) This compares to only 34 minutes for shape from motion[1] to collect up to 500 measurement vectors using the single 33.71 N weight. Although the time difference is not as dramatic as the 2-DOF case, it is significant and gets more so as the number of least squares force vectors increases. Also, these times are aided by the use of the calibration fixture, which makes applying precise loads easier, primarily benefiting least squares.

Gathering the raw data for the shape from motion trials involved sampling the sensor at 2 Hz while it slowly moved up and down the longitudinal lines of an imaginary sphere. Each trial varied slightly in length so the total volume of data was different each time, but the raw data collection time was a small percentage (10-15%) of the total. The majority of time was consumed in gathering the precise loads used to orient the calibration matrix with the desired reference frame. This was done to the same precision as the least squares loads.

To compare precision, we performed the same experiments as in the previous subsection but randomly moved the sensor in 3-D rather than just the plane. The extremes of these data are tabulated in Table 2. Again, shape from motion accuracy (mean) is as good or better while imprecision (standard deviation) is always better.

1. both include the time to compute applied force vectors

To verify the absolute accuracy of both techniques, we assembled the calibration fixture in two configurations that were not used in either calibration procedure and gathered data in ten different orientations. The accuracy of each applied load in orientation and force is 1 milliradian and 0.02 Newtons, respectively. Table 3 shows average errors in force magnitude and direction across all ten trials.

7. DISCUSSION

We presented our motivation for learning sensorimotor primitives by outlining our concept of a gesture-based programming paradigm. Gesture-Based Programming is a form of programming by human demonstration that requires a knowledge base of sensorimotor primitives which encode elements of autonomous sensorimotor dexterity that can be assembled into robotic skills. Another requirement of the paradigm is a mechanism to recognize the primitives and combine them.

In support of this, we described shape from motion primordial learning, a learning technique based on principle components analysis that has been successfully applied to extracting and identifying robotic sensorimotor primitives from teleoperated actions on both a mobile robot and a manipulator. By incorporating model-based information, we have also applied it to nearly autonomous calibration of force/torque sensors with very good results.

For the mobile robot, the technique learned to escape from blocking obstacles, front and back, and wheel jams. It learned this with no explicit knowledge of its own sensors and actuators. It merely extracted the most significant mappings between inputs and outputs during a teleoperated training phase by an expert human and developed a linearized model that satisfactorily mimicked the human's behavior.

Likewise, for the PUMA manipulator, sensorimotor primitives representing guarded moves and edge accommodation were extracted from teleoperated trials. Essentially, the robot was learning various configurations of damping force controllers. In the case of edge accommodation, the primitives were successfully extracted in the presence of the more dominant guarded move primitive. In these experiments, the guarded move was known to exist in the dataset *a priori*, but we further demonstrated that this eigenspace representation allows the automatic recognition and extraction of previously learned primitives, allowing the system to seek out new underlying primitives during the learning process.

As mentioned above, the ability to recognize previously learned primitives is important not only for seeking out new primitives but also for understanding tasks during human demonstration. We demonstrated the applicability of this approach to the GBP paradigm by using it to recognize primitives from actual human demonstrations as well as teleoperations.

Of course, this approach has drawbacks, as well. The biggest problem stems from its greatest strength: linearity. Linearity provides for convenient and compact representation with convenient parallelism metrics and superposition. This is what allows primitive recognition and primitive "morphing." (The combination and transformation of primitives into new primitives.) So far, we have only attempted to learn primitives that lend themselves well to linear representations. But most of the world is non-linear so it remains to be seen if this approach is useful for more complex tasks. The good news is that our approach is intrinsically segmented, so we do not have to linearize the whole "world." We only need to determine piecewise linear segments of the world. This is not a trivial problem, but Kang and Ikeuchi have already done some intelligent segmentation for Robotic Instruction by Human Demonstration [11] and we are building on that foundation. While other representations such as neural networks can build nonlinear models, making the segmentation problem easier, they do not eliminate the problem and they make the recognition and morphing problems much more difficult. We believe, for now, our approach yields a more promising compromise.

8. ACKNOWLEDGEMENTS

Mr. Voyles was supported in part by a National Defense Science and Engineering Graduate Fellowship and by a DOE Integrated Manufacturing Predoctoral Fellowship. Mr. Morrow was supported in part by a DOE Computational Science Graduate Fellowship and by a DOE Integrated Manufacturing Predoctoral Fellowship.

9. REFERENCES

[1] Bicchi, A. and P. Dario, 1988, "Intrinsic Tactile Sensing for Artificial Hands," *Robotics Research: The 4th International Symposium*, R.C. Bolles and B. Roth, editors, MIT Press, Cambridge, MA, pp. 83-90.

[2] Brooks, R.A., "A Robust Layered Control System for a Mobile Robot," *IEEE Journal of Robotics and Automation*, v.RA-2, n.1, March 1986, pp. 14-23.

[3] Gertz, M.W, *A Visual Programming Environment for Real-Time Control Systems*, Ph.D. Thesis, Department of Electrical and Computer Engineering, Carnegie Mellon University, 1994.

[4] Gertz, M.W., D.B. Stewart and P.K. Khosla, "A Software Architecture-Based Human-Machine Interface for Reconfigurable Sensor-Based Control Systems," in *Proc. of the 8th IEEE Symp. on Intelligent Control*, Chicago, IL, August 1993.

[5] V. Gullapalli, R. Grupen, and A. Barto, "Learning Reactive Admittance Control," in *Proceedings of IEEE International l Conference. on Robotics and Automation*, Nice, France, 1992.

[6] Hancock, J. and C. Thorpe, "ELVIS: Eigenvectors for Land Vehicle Image System," in *Proc. of 1995 IEEE/RSJ International Conf. on Intelligent Robots and Systems*," Pittsburgh, PA, Aug., 1995, pp 35-40.

[7] Hartley, J., *Robots at Work*, IFS Publications, Ltd.,Kempston, Bedford, UK, 1983, chapters 5, 8.

[8] Hebb, D.O., "The Organization of Behavior; a Neuropsychological Theory," New York, Wiley, 1949.

[9] Kang, S.B., *Robot Instruction by Human Demonstration*, Ph.D. Thesis, The Robotics Institute, Carnegie Mellon University, CMU-RI-TR-94-33, November, 1994.

[10] Kang, S.B. and K. Ikeuchi, "Robot Task Programming by Human Demonstration: Mapping Human Grasps to Manipulator Grasps," in *Proc. of IEEE/RSJ/GI International Conference on Intelligent Robots and Systems*, Munchen, germany, 1994, pp. 97-104.

[11] Kang, S.B., and K. Ikeuchi, "Toward Automatic Robot Instruction from Perception -- Temporal Segmentation of Tasks from Human Hand Motion," in *IEEE Transactions on Robotics and Automation*, v., n. Mar. 1996.

[12] Kang, S.B., J. Webb, C.L. Zitnick, and T. Kanade, "An Active Multibaseline Stereo System with Real-Time Image Acquisition," in *Proc. of Image Understanding Workshop*, Monterey, CA, Nov., 1994.

[13] Kuniyoshi, T., M. Inaba, and H. Inoue, "Teaching by Showing: Generating Robot Programs by Visual Observation of Human Performance," in *Proceedings of the 20th International Symposium on Industrial Robots*, 1989, pp. 119-126.

[14] S. Lee and M. H. Kim, "Learning Expert Systems for Robot Fine Motion Control," pp. 534-544, *Proceedings of the IEEE International Symposium on Intelligent Control*, Arlington, VA, 1988.

[15] S. Liu and H. Asada, "Transferring Manipulative Skills to Robots: Representation and Acquisition of Tool Manipulative Skills Using a Process Dynamics Model," *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 114, No. 2, pp. 200-228, June 1992.

[16] Maes, P. and R. Brooks, "Learning to Coordinate Behaviors," AAAI-90, Boston, MA, pp. 796-802, 1990.

[17] P. Michelman and P. Allen, "Forming Complex Dextrous Manipulations from Task Primitives," pp. 3383-3388, *Proceedings of IEEE International Conference on Robotics and Automation*, San Diego, CA, 1994.

[18] Morrow, J.D. and P.K. Khosla, "Sensorimotor Primitives for Robotic Assembly Skills," in *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, Nagoya, Japan, May, 1995.

[19] Patrick, J., *Training: Research and Practice*, Academic Press, San Diego, CA, 1992.

[20] W. Paetsch, and G. von Wichert, "Solving Insertion Tasks with a Multifingered Gripper by Fumbling," pp. 173-179, *Proceedings of IEEE International Conference on Robotics and Automation*, Atlanta, GA, 1993.

[21] D. Pomerleau, *Neural Network Perception for Mobile Robot Guidance*, Ph.D. Thesis, Carnegie Mellon University, Pittsburgh, PA, 1992.

[22] Pierce, D., "Learning a Set of Primitive Actions with an Uninterpreted Sensorimotor Apparatus," 8th International Workshop on Machine Learning, Evanston, IL, 1991, pp 338 - 342.

[23] Pook, P.K. and D.H. Ballard, "Recognizing Teleoperated Manipulations," in *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*, v. 2, pp. 578-585, May 1993.

[24] K. Shimokura and S. Liu, "Programming Deburring Robots Based on Human Demonstration with Direct Burr Size Measurement," pp. 572-577, *Proceedings of IEEE International Conference on Robotics and Automation*, San Diego, CA, 1994.

[25] J. Simons, H. Van Brussel, J. De Schutter, and J. Verhaert, "A Self-Learning Automaton with Variable Resolution for High Precision Assembly by Industrial Robots," *IEEE Transactions on Automatic Control*, Vol AC-27, No. 5, October, 1982.

[26] T.H. Speeter, "Primitive-Based Control of the Utah/MIT Dextrous Hand," *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 866-877, Sacramento, CA, April 1991.

[27] Tomasi, C. and T. Kanade, 1991, "Shape and Motion from Image Streams: a Factorization Method" Tech. Report CMU-CS-91-172, Computer Science, Carnegie Mellon University, Pittsburgh, PA.

[28] Stewart, D.B., R.A. Volpe, and P.K. Khosla, "Design of Dynamically Reconfigurable Real-Time Software Using Port-Based Objects," CMU Robotics Institute tech. report, CMU-RI-TR-93-11, July, 1993.

[29] Todd, D.J., Fundamentals of Robot Technology, John Wiley and Sons, 1986, chapters 3, 7.

[30] Tomovic, R., "Transfer of Motor Skills to Machines," in *Robotics and Computer-Integrated Manufacturing*, v. 5, n. 2/3, 1989, pp. 261-267.

[31] Uchiyama, M., E. Bayo, and E. Palma-Villalon, 1991, "A Systematic Design Procedure to Minimize a Performance Index for Robot Force Sensors," *Journal of Dynamic Systems, Measurement, and Control*, v.113, n.3, pp. 388-394.

[32] E.G. Vaaler and W.P. Seering, "A Machine Learning Algorithm for Automated Assembly," *Proceedings of IEEE International Conference on Robotics and Automation*, 1991.

[33] Voyles, R.M., G. Fedder, and P.K. Khosla, "Design of a Modular Tactile Sensor and Actuator Based on an Electrorheological Gel," *Proceedings of IEEE International Conf. on Robotics and Automation*, 1996.

[34] Voyles, R.M. and P.K. Khosla, (1995a). "Tactile Gestures for Human/Robot Interaction," *Proc. of IEEE/RSJ Conf. on Intelligent Robots and Systems*, Pittsburgh, PA, v. 3, pp. 7-13.

[35] Voyles, R.M. and P.K. Khosla, (1995b). "Multi-Agent Gesture Interpretation for Robotic Cable Harnessing," *Proc. of IEEE Conf. on Systems, Man, and Cybernetics*, Vancouver, B.C., pp. 1113-1118.

[36] Voyles, R.M., J.D. Morrow, and P.K. Khosla, 1995, "Shape from Motion Decomposition as a Learning Approach for Autonomous Agents," *Proceedings of the IEEE International Conf. on Systems, Man, and Cybernetics*, v. 1, pp 407-412.

[37] Voyles, R.M., J.D. Morrow, and P.K. Khosla, "The Shape from Motion Approach to Rapid and Precise Force/Torque Sensor Calibration," *Journal of Dynamic Systems, Measurement and Control*, June, 1997.

[38] J. Yang, Y. Xu, and C.S. Chen, "Hidden Markov Model Approach to Skill Learning and Its Application to Telerobotics," *IEEE Transactions on Robotics and Automation*, Vol. 10, No. 5, pp. 621-631, October 1994.

[39] D.S. Ahn, H.S. Cho, K. Ide, F. Miyazaki, S. Arimoto, "Strategy Generation an Skill Acquisition for Automated Robotic Assembly Task, Proceedings of the IEEE ISIC, 1991, pp. 128-133.

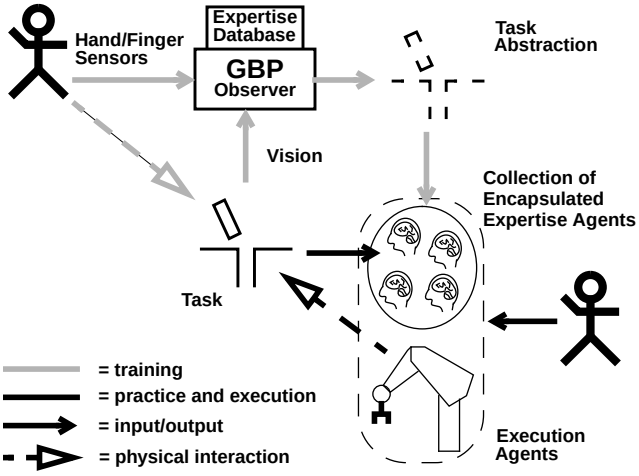

FIGURE 1: GESTURE-BASED PROGRAMMING (GBP) SYSTEM OVERVIEW

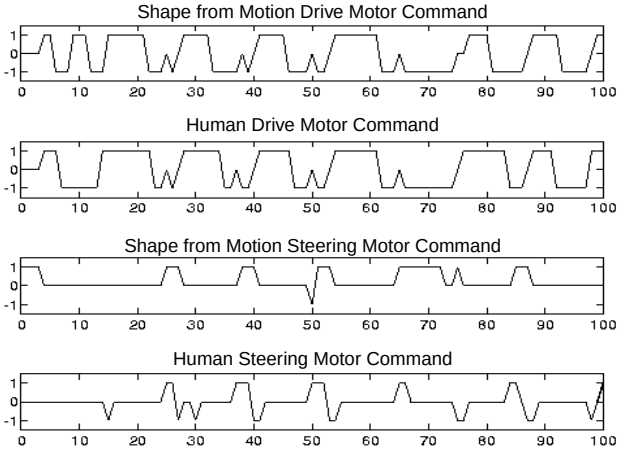

FIGURE 2: COMPARISON OF HUMAN AND SHAPE FROM MOTION COMMANDS ON A TELEOPERATED

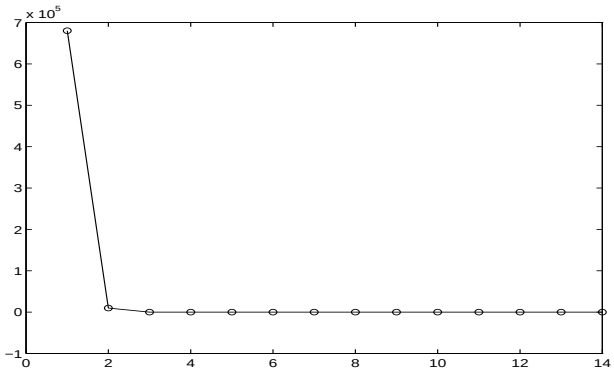

FIGURE 3: SINGULAR VALUES ASSOCIATED WITH THE FIRST 14 EIGENVECTORS OF THE GUARDED MOVE TRAINING DATA.

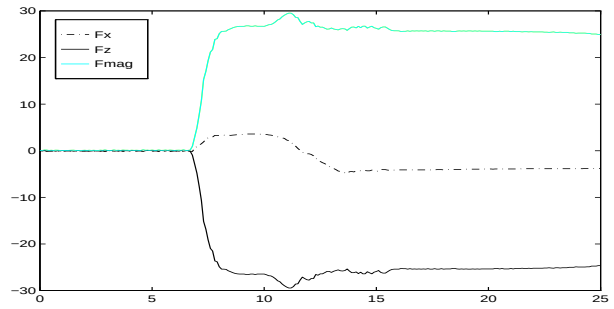

FIGURE 4: AUTONOMOUS OPERATION OF THE LEARNED, ONE-DIMENSIONAL GUARDED MOVE.

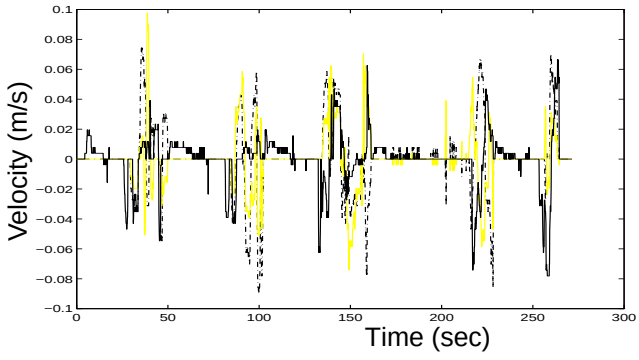

FIGURE 5: CARTESIAN VELOCITY COMMANDS RECORDED DURING RANDOM TELEOPERATED MOTION AND INTERACTIONS WITH OBJECTS.

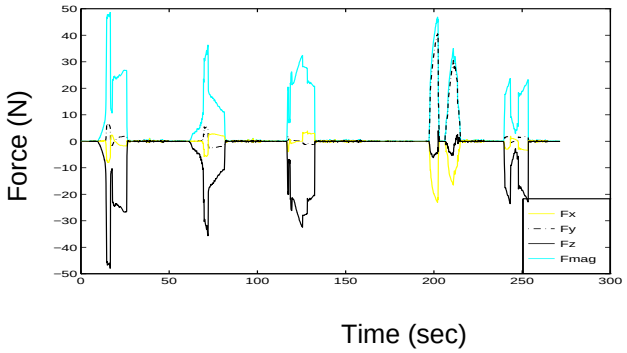

FIGURE 6: FORCE COMPONENTS RECORDED DURING RANDOM TELEOPERATED MOTION AND

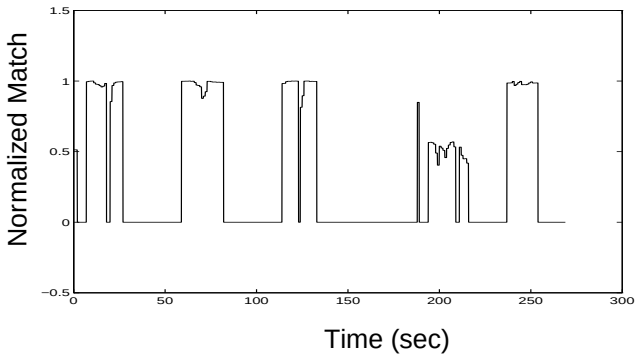

FIGURE 7: GOODNESS OF MATCH DETERMINED BY AUTOMATIC SKILL IDENTIFICATION ALGORITHM DURING RANDOM TELEOPERATED MOTION AND INTERACTION WITH OBJECTS.

.

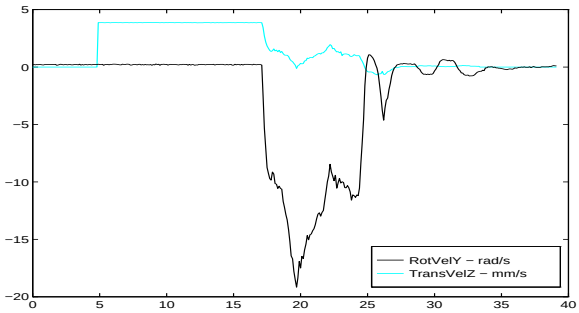

FIGURE 8: VELOCITY COMMANDS OF “YROLL” AND “ZGUARD” PRIMITIVES WORKING COOPERATIVELY

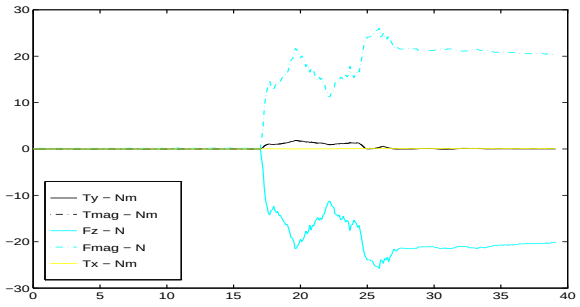

FIGURE 9: MEASURED FORCES WITH “YROLL” AND “ZGUARD” PRIMITIVES WORKING COOPERATIVELY TO

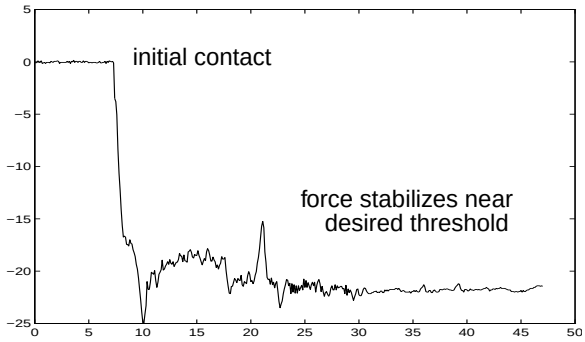

FIGURE 10: FORCE IN Z-AXIS AS ZGUARD, XROLL, AND YROLL COOPERATE TO PRESS ONE SURFACE AGAINST ANOTHER.

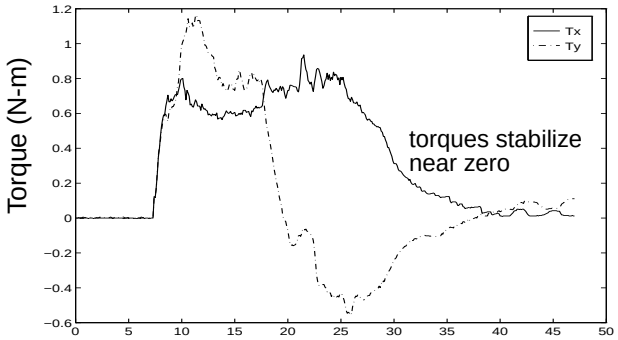

FIGURE 11: TORQUE AROUND X- AND Y-AXES AS ZGUARD, XROLL, AND YROLL COOPERATE TO PRESS ONE SURFACE AGAINST ANOTHER.

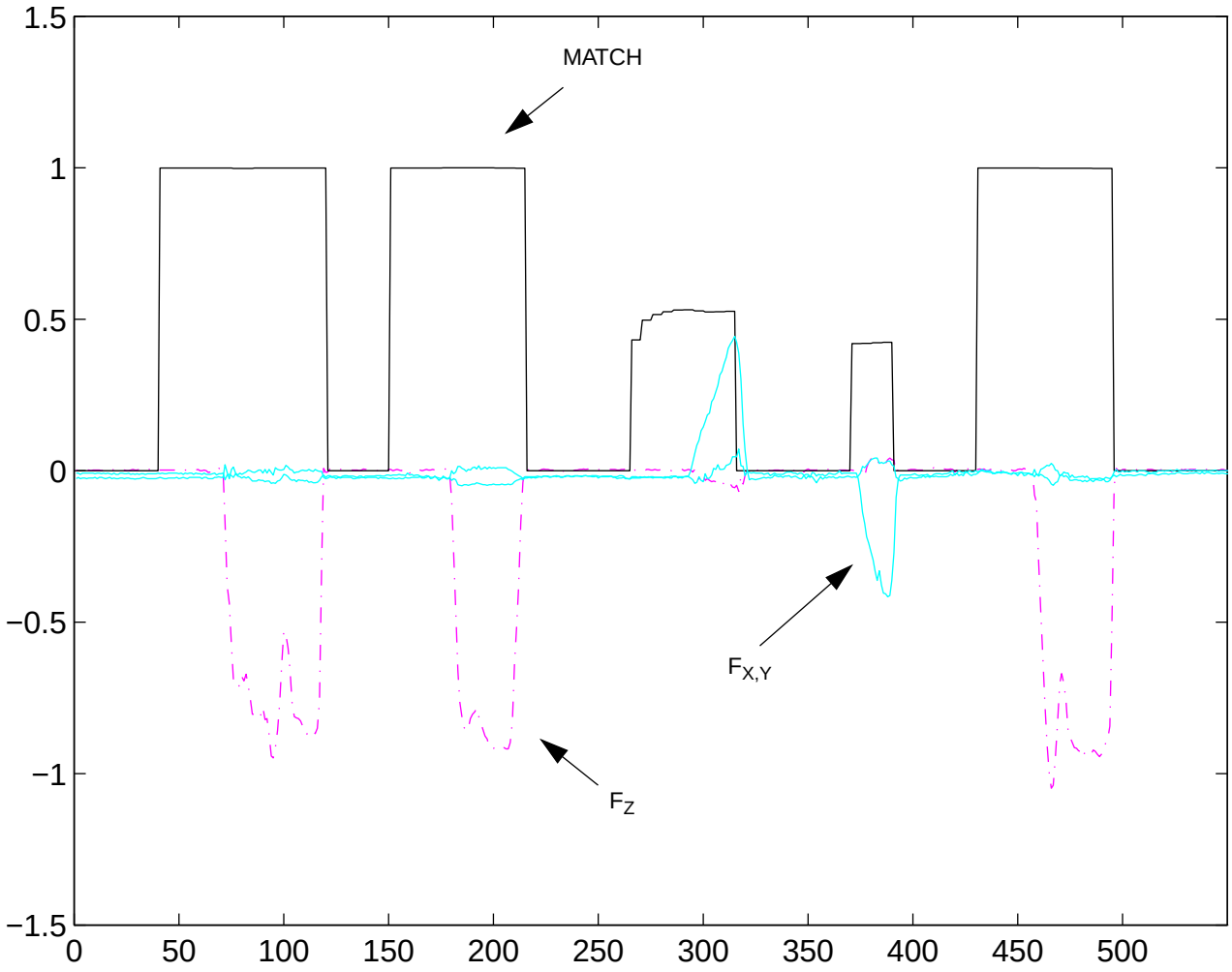

FIGURE 12: RECOGNITION OF ZGUARD PRIMITIVE FROM HUMAN DEMONSTRATION

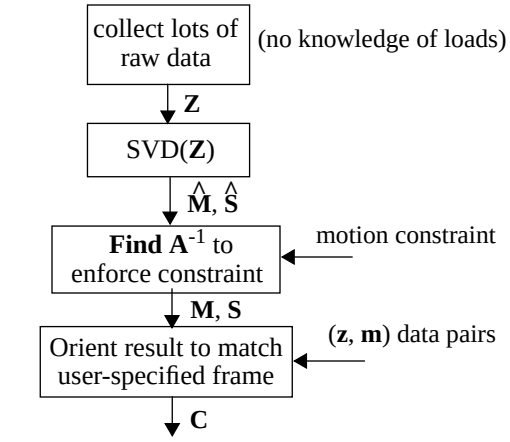

FIGURE 13: SHAPE FROM MOTION CALIBRATION PROCEDURE

FIGURE 14: THE FINGERTIP SENSOR

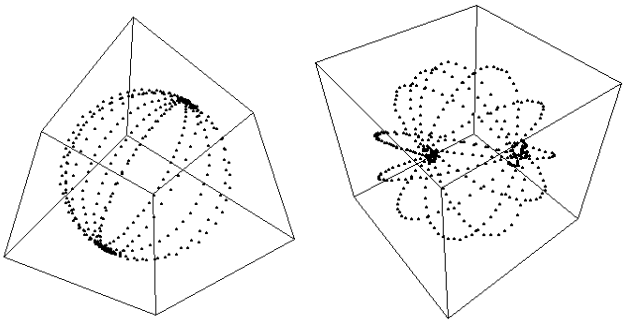

FIGURE 15: RECOVERED MOTION OF LORD F/T SENSOR.

Table 1: FINGERTIP SENSOR COMPARISON

	Least Squares				Shape from Motion			
load (g)	mean (g)	error (%)	imprecision (std dev)	time (min)	mean (g)	error (%)	imprecision (std dev)	time (min)
60.4	60.1	0.5	1.20	17	60.5	0.2	1.13	1.5
111.4	110.2	1.1	0.83		111.6	0.2	0.60	

Table 2: LORD F/T SENSOR COMPARISON

	Least Squares				Shape from Motion			
load (N)	mean (N)	error (%)	imprecision (std dev)	time (min)	mean (N)	error (%)	imprecision (std dev)	time (min)
9.26	9.09	1.8	.084	69	9.09	1.8	.058	34
25.24	25.53	1.7	.157		25.41	0.7	.062	

Table 3: AVERAGE LORD FORCE ERRORS

method	magnitude error (N)	angle error (rad)
Shape from Motion	0.080	0.0025
Least Squares	0.144	0.0046