The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Wiimote Robot Control Using Human Motion Models

Christian Smith* and Henrik I Christensen†
* Centre for Autonomous Systems, Royal Institute of Technology
Stockholm, Sweden. Email: ccs@kth.se
†Robotics and Intelligent Machines, Georgia Institute of Technology
Atlanta, GA 30332-0760. Email: hic@cc.gatech.edu

Abstract— **As mass-market video game controllers have become more advanced, there has been a recent increase in interest for using these as intuitive and inexpensive control devices. In this paper we examine position control for a robot using a wiimote game controller. We show that human motion models can be used to achieve better precision than traditional tracking approaches, sufficient for simpler tasks. We also present an experiment that shows that very intuitive control can be achieved, as novice subjects can control a robot arm through simple tasks after just a few minutes of practice and minimal instructions.**

I. INTRODUCTION

As robots become increasingly prolific, the number of people that come into contact with them grows steadily. From having been specialized tools used by trained specialists, more and more effort is put into making robots into general tools used by untrained operators. Therefore, it is important to find intuitive modes of interaction that require as little training as possible of the users. Also, the less effort the user has to put into operating the robot, the more effort can be put into solving the task at hand.

In recent years, mainstream computer game interfaces have become more and more advanced and are beginning to receive more interest as tools for robot interaction. By their very design, these interfaces are meant to be used by untrained laymen, while still enabling detailed enough control to keep games interesting.

As reported in [1]–[6], the remote control device for the Nintendo Wii video game, the accelerometer-equipped "wiimote" interface is intuitive and easy to use. Even though it might not be as accurate as a traditional control interface, the cognitive load is lower, allowing the user to concentrate on other things, or performing several tasks simultaneously. Others report that wiimote control is less precise and lowers task performance as compared to more traditional input devices [7]. Recently, an industrial robot arm controlled with a wiimote has attracted media attention [8], but so far this has only been using task level control where the task is not performed until the user input motion has ended. To the knowledge of the authors, there are no reports of direct position tracking using the wiimote accelerometers.

As the wiimote and other gaming interfaces are intended for a mass market, they are cheap and easily available. A drawback, however, is that the precision in the motion detection hardware leaves a lot to be desired for high-precision industrial or scientific applications.

In the present paper, we present a way to use human motion models to accurately track human input using the wiimote controller. With the presented approach using minimum jerk trajectories, position control to within a few centimeters of a desired target is attainable. The method enables simultaneous — and for some cases even predictive — tracking, so the input can be analyzed in real time, making it possible to implement task level control that performs the task simultaneously with the user's input.

The paper is structured in the following way: Section II describes the motion models used, Section III describes the technical details of the implementation, Sections IV and V describe experiments used to verify the approach, and the conclusions are presented in Section VI.

II. MODEL-BASED CONTROL

In this section we discuss the need for model based control, present the minimum jerk model, and argue why this is suitable for the present problem.

A. Position Tracking with Accelerometers

The wiimote uses MEMS accelerometers. Consequently, the accuracy is limited and one has to consider the challenges of using sensors that have limited accuracy.

The first challenge is one of observability. In the strict sense of the word, position is not observable from acceleration measurements alone, as one will not know the initial position or velocity. In practice, however, one can assume that a motion starts at rest at the origin, adding initial conditions that resolve observability.

The second problem is more difficult. A measurement $\hat{\ddot{x}}$ of acceleration $\ddot{x}$ will be corrupted by noise w. Thus, when we integrate $\hat{\ddot{x}}$ twice to get an estimate $\hat{x}$ of position x, we will have that $\hat{x} \sim x + wt^2$. That is, the error is now proportional to t^2, and will tend to infinity over time. In practical cases, just taking the naïve double integral of acceleration measurements from a wiimote will give position estimates where the noise to signal ratios exceed 1 after just a few seconds. This is a well-known problem, and has been partially solved for shorter durations by removing bias errors and filtering the results, with e.g. a Kalman filter [9], [10]. However, these approaches only minimize the error coefficient w, they do not eliminate the t^2 dependency. Due to these problems, up to now, accelerometers have mainly been used for qualitative input, such as gesture recognition based interfaces [4], [11].

B. Minimum Jerk Motion Model

We propose to use human motion models to cope with the aforementioned tracking problems. We assume a parametrized model for possible motions and estimate the parameters from measurements, thus guaranteeing that the generated motion belongs to a subset of valid motions.

A well-known model for explaining the kinematics of visually guided human reaching motions is the *minimum jerk* (MJ) model. It was first proposed by Hogan for single-joint motions in [12], and later extended to include multi-joint planar motion in [13]. More recently, other, more detailed models like minimum joint torque change, minimum force, or minimum energy have been proposed and shown to describe human motions more accurately [14], [15].

However, the more detailed models require detailed knowledge of posture and intrinsic mechanical parameters of the human subject, parameters that are not observable when using a handheld wiimote for motion input. On the other hand, as we shall see below, the MJ model can be completely described in cartesian space coordinates with no explicit knowledge of the subject's intrinsic mechanical properties, and is thus possible to implement in our scenario. Furthermore, when a subject is performing free motion of moderate extents with the wiimote, it is reasonable to assume that no external forces other than gravity act on the user, and that posture does not change significantly during the motion. In this case, it has been shown that trajectories predicted by the MJ model do not differ significantly from ones predicted by the more advanced models [14].

MJ theory is based on the observation that the trajectory of voluntary arm motions when described in extra-corpular cartesian space, can be described with a model in which the square sum of the third derivative of position, *jerk*, integrated over time is minimized [12], [13]. Given a starting point, an end point and a time to move between the two, the trajectory that minimizes the jerk on this interval is the MJ trajectory. All MJ trajectories share the property that the 6th derivative is zero for the duration of the motion, and that they thus can be described as 5th degree polynomials, as in Equation 1.

$$\mathbf{x}(t) = a_1t^5 + a_2t^4 + a_3t^3 + a_4t^2 + a_5t + a_6 \quad (1)$$

If we also add the start and end points of the motion, $\mathbf{x}(t_0)$ and $\mathbf{x}(t_1)$, and state the position, velocity, and acceleration at these points, we get the following constraints on Equation 1.

$$\begin{aligned} \mathbf{x}(t_0) &= \mathbf{x_0}, & \mathbf{x}(t_1) &= \mathbf{x_1} \\ \dot{\mathbf{x}}(t_0) &= \dot{\mathbf{x}}_\mathbf{0}, & \dot{\mathbf{x}}(t_1) &= \dot{\mathbf{x}}_\mathbf{1} \\ \ddot{\mathbf{x}}(t_0) &= \ddot{\mathbf{x}}_\mathbf{0}, & \ddot{\mathbf{x}}(t_1) &= \ddot{\mathbf{x}}_\mathbf{1} \end{aligned}$$

The above constraints give us 6 equations, and we get a well-defined system to find the 6 parameters $a_1 \dots a_6$. Thus, there is only one possible MJ trajectory for a given start and end, and it can be found by solving a simple system of linear equations. This is illustrated with an example of a human's reaching motion recorded from an experiment in [16] with the MJ trajectory superimposed, in Fig. 1.

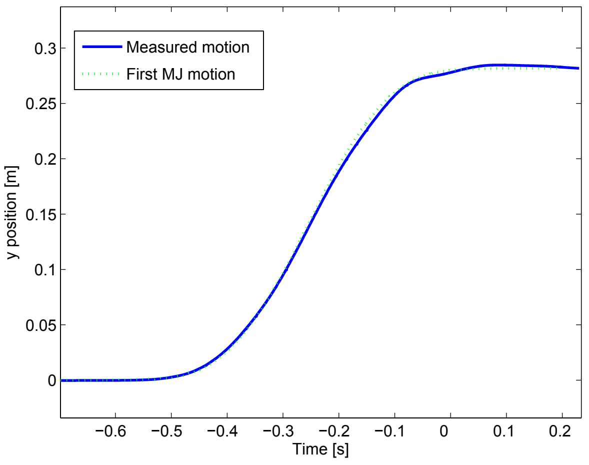


Fig. 1. Y component of measured hand trajectory with MJ trajectory fitted. In this case the hand trajectory contains only one major MJ component.

C. Model-based Input Tracking

Using the assumption that the user only performs isolated reaching motions, an input tracking system based on MJ theory was designed. Here, "isolated" is used in the sense that one motion does not start until the previous motion is ended, in practice requiring a 100 ms separation. Thus, the constraints are set so that $\dot{x}_0$, $\ddot{x}_0$, $\dot{x}_1$, and $\ddot{x}_1$ are all zero, and each motion is assumed to start where the last motion ended. This ensures that all motions end at zero velocity, eliminating buildup error. Since the measurements are in acceleration space, Equation 1 was differentiated twice to get Equation 2.

$$\ddot{\mathbf{x}}(t) = 20a_1t^3 + 12a_2t^2 + 6a_3t + 2a_4 \quad (2)$$

We assume that all motions start and end with zero acceleration, which gives the following constraints:

$$\ddot{\mathbf{x}}(t_0) = 0, \qquad \ddot{\mathbf{x}}(t_1) = 0$$

Then, Equation 2 can be rewritten parametrized by the single constant a_1, as in Equation 3.

$$\begin{aligned} \ddot{\mathbf{x}}(t) = \; & a_1(20t^3 - 30(t_0+t_1)t^2 \\ & +10(t_0^2+t_1^2+4t_0t_1)t \\ & -10(t_0^2t_1+t_0t_1^2)) \end{aligned} \quad (3)$$

The remaining constants are found as:

$$\begin{aligned} a_2 &= -\tfrac{5}{2}(t_0+t_1)a_1 \\ a_3 &= \tfrac{5}{3}(t_0^2+t_1^2+4t_0t_1)a_1 \\ a_4 &= -5(t_0^2t_1+t_0t_1^2)a_1 \end{aligned}$$

Given a consecutive stream of acceleration measurements, Equation 3 can be fitted to a segment of these using least squares fitting. The initial state is assumed to be non-moving, and all acceleration measurements beneath a certain threshold are ignored. When four consecutive measurements above the threshold have been recorded, the fitting process is initialized. Thus, in the beginning of a motion, only a small part of the curve will be fit to actual measured points — the rest will be extrapolated.

Two of the parameters in Equation 3 are the unknown start and end times of the motion, t_0 and t_1. These are found using an exhaustive search. Results in [16] show that we should expect motion durations from 0.4 s to 1.0 s, so we search an interval from 0.3 s to 1.3 s to cover all expected durations. We call the time where the acceleration threshold is exceeded t_{tr}. We then let t_0 take all values from $t_{tr} - 100$ ms to $t_{tr} + 100$ ms in 10 ms steps, and for each value of y_0, we let t_1 take the values from $t_0 + 300$ ms to $t_0 + 1000$ ms, also in 10 ms intervals. For each interval $[t_0, t_1]$, we find the least square fit using all available data for the interval. The average residual error is calculated for each possible interval by dividing the residual sum with the number of measurement points, and the one with the smallest average residual is chosen as the interval and trajectory to use.

Eventually, a more efficient way to find the start and end points should be implemented, but as at most approximately 400 datapoints will be used for least squares fitting for 1400 possible time intervals, the trajectory fitting is easily run in a few milliseconds, allowing realtime implementation on a regular desktop computer. Therefore, this has not been an urgent point of improvement.

The average residual error is summed over all three dimensions, so large distinct minimum jerk type motions in one or more dimensions will take precedence over noisier, less distinct motions in the remaining dimension(s). As it is the same motion, it should start and end simultaneously in all dimensions, and any deviation from this can be viewed as noise and ignored. In order to get the trajectory in position space, the polynomial is integrated, using zero initial velocity and the last known position as the constants of integration. For an illustration of acceleration measurements and the fit function, see Figure 2.

When the endpoint t_1 has been passed, the system will not attempt to detect the next motion until 100 ms have passed, in order to avoid treating the end of one motion as the beginning of the next.

In this control mode, the current velocity and position obtained from integrating the polynomial one or two times respectively can be sent directly to the robot controller for real-time control. As time progresses, more and more points will be used for fitting the curve, and thus the precision improves as the trajectory is executed. As was shown in [17], the trajectory is stable enough that further measurements have little impact after about half of the motion has been performed. Since the endpoint of the polynomial trajectory is trivial to find, this means that a decent approximation of the endpoint of the hand trajectory is available after half of the motion has been performed.

The limitations of this approach is that it is only possible to track (series of) isolated motions. It is not suited for following complicated continous paths, as 100 ms of motion would be cut out between each detected MJ trajectory, resulting in severe drift as the robot moves less than the user.

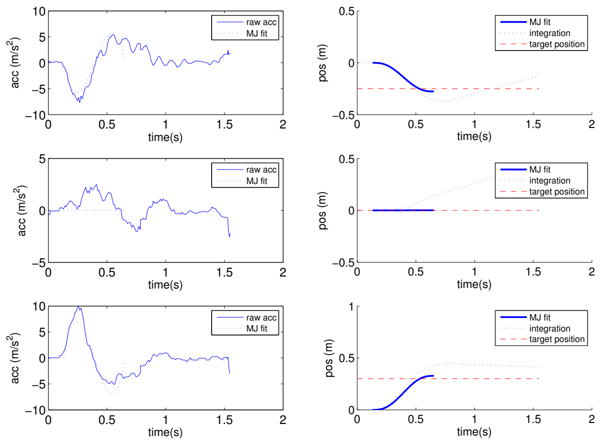


Fig. 2. The left figure shows the third degree curve fit to measured acceleration. The right plot shows the same data integrated to position space. The dashed horizontal lines in the right figure show the positions of the target.

III. Implementation

The input tracking algorithm described in Section II-C was implemented and evaluated on motion data using a wiimote controller.

A. User Interface

The user interface consists of a stock wiimote video game controller, see Figure 3. The features that are used in this setup are 3 linear accelerometers with a range up to ± 50 m/s^2 and 0.38 m/s^2 resolution, a 1024×768 pixel IR camera, and a trigger button. User input is read at 100 Hz. The accelerometers are reported to have an accuracy of $\pm 10\%$ [1], but this has not been verified in the present setup. The accelerometers were calibrated so that constant and linear bias errors were eliminated.

Fig. 3. The Wiimote input device.

An IR LED setup provided with the wiimote is used with the camera to determine the orientation when the user is pointing forward. When the LEDs are not visible, an EKF keeps track of the orientation, also making use of the accelerometer data generated by gravity whenever the wiimote is assumed to be nearly motionless. When the LEDs are not visible, the orientation measurements become slightly less accurate, but the main difference to using LED data is that rotation can not be measured along the vertical axis, limiting tracking to 5 DoF instead of 6. Apart from tracking the direction the user is pointing, the orientation data is also used to subtract the Earth's gravity from the accelerometer readings, as well as to convert measurements from the wiimote frame of reference to the global frame of reference. The trigger button is connected to a reset function that sets position and velocity to zero, used to initialize the reference zero position.

B. *Evaluation setup*

The system was evaluated with a series of simple experiments. For these experiments, a ground truth measure of the position was obtained using a simple motion capture setup consisting of a stereo camera pair that tracked a marker at the base of the wiimote. The stereo pair has a 60 cm baseline. The tracking error is less than 1 cm in the workspace used, and temporal resolution is 50 Hz.

IV. Evaluation Experiments

A series of trials were carried out to evaluate the performance of the MJ based tracking method.

A. *Proof of Concept*

In the first trial, we demonstrate proof of concept by moving the wiimote freely in space through a series of distinct reaching motions. The motion capture data was then compared with the trajectories generated by the MJ estimator. The resulting trajectories are shown in Figure 4. These results are illustrative of the MJ model based tracking performance. In comparison, the performance when simply taking the double integral of the accelerometer readings on the same data does no longer fit in the plot after the first few seconds, but the accumulated position error is 62.7 m and the velocity error is 3.18 m/s at the end of the 37 second session.

B. *Tracking Experiment*

We conducted a tracking experiment where we collect statistics on tracking performance and compare these to other more straightforward methods, in order to achieve a quantitative measure of precision.

In this experiment, a set of physical targets were set up. These consist of colored plastic balls that can be touched with the wiimote. One ball was designated as a starting point, and three other balls were mounted on a line 0.3 m above it, see Figure 5. Subjects were asked to start by touching the starting point ball, and then touch the colored balls as called out by the experimenter. The starting point was located at about chest height in front of the subject, and the other targets

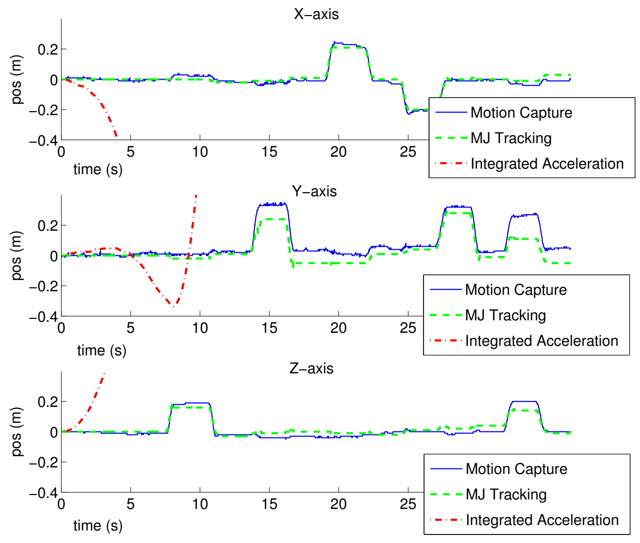


Fig. 4. The trajectory of the wiimote estimated with the MJ approach compared to the trajectory recorded by a motion capture system.

were located approximately at face height. Each subject was asked to carry out 40 such touching motions, according to a predetermined pattern. In total, 5 subjects were used. They were not given any other instructions than to start stationary and touch the colored balls as called out, and had not been told the purpose of the experiment or the details of the tracking system before the experiment.

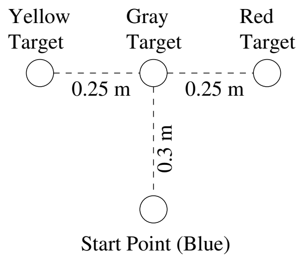


Fig. 5. Schematic of targets used in tracking experiment. Subjects start with the wiimote touching the start point, and move the wiimote to touch the colored targets as called out by the experimenter.

The position as tracked by the motion capture system, the position as tracked by the MJ based tracker, and all raw measurement data were logged. For comparability, each motion was analyzed separately, rather than measuring the total error after all motions were completed.

The results of the MJ based tracker were compared to the motion capture positions, and the error, expressed as the distance between the two positions at the end of each motion when the wiimote had stopped at the target ball, was recorded. For a baseline of comparison, the raw data logged from the wiimote was parsed offline through two alternative tracking methods.

First, the result obtained when performing a naïve double integration of the acceleration was calculated. In this case, position and velocity was reset at the beginning of each motion, to attenuate buildup error. Motion start and end times manually picked from the motion capture tracker were used for limits of integration. This minimized the buildup error by excluding data not from the actual motion duration.

A more refined approach is to use a Kalman filter for the tracker. The same acceleration data as with the naïve integrating approach was used, and the filter parameters were tuned manually offline in order to minimize the error on this dataset. This guarantees that we achieve the best possible Kalman filter results, and minimize the negative impact of filter tuning choices. In an online implementation, a Kalman filter would perform worse than this.

The resulting errors at the motion endpoints are summarized in Table I. As can be seen, the MJ based approach outperforms the other two for precision. This is so even though the MJ based approach detects motion start and endpoints automatically. This detection has to be done manually for the other systems, to prevent small errors in acceleration to build up to considerable drift in velocity and position.

TABLE I

RESULTS FROM REACHING EXPERIMENT

Method	avg error	std error
Naïve integration	0.097 m	0.108 m
Kalman Filter	0.074 m	0.082 m
MJ model	0.061 m	0.051 m

V. ROBOT CONTROL EXPERIMENT

The previous section showed that an MJ based tracker can follow human motions more accurately than traditional tracking approaches. To test the feasability of using this type of input device for control of a robot manipulator, a simple user experiment was carried out. In this experiment, the subjects used the wiimote to steer a robot to touch colored balls in a task similar to the one in the previous section.

A. Robot Platform

The robot used in this experiment is a fast and lightweight manipulator with six degrees of freedom, see Figure 6. If started in the center, the robot can reach any point in its 60 cm $\times$ 60 cm workspace within 0.55 s, with the settings used in this experiment. The end effector used in this experiment is a floorball mounted on a 10 cm piece of PVC pipe. A detailed description of the manipulator can be found in [18].

B. Procedure

The colored ball targets described in Section IV-B were placed so that they were reachable by the robot. The target balls were mounted on PVC pipes so that they were situated at 25 cm intervals 30 cm above the robots centered default position. The targets will move when hit, but come att rest in the original position within a few seconds. See Figure 6 for an illustration of the setup.

The subjects used the robot to carry out the same target-touching tasks that were done directly with the wiimote in the previous experiment. They were given a brief instruction in how the control system worked, and were asked to hold the wiimote so that its position parallelled the position of the robot. Before the experiment started, they were given a few minutes free practice to become acquainted with controlling the robot. In total, 15 subjects were used, 12 male and 3 female. All subjects were right-handed and had normal or corrected to normal vision.

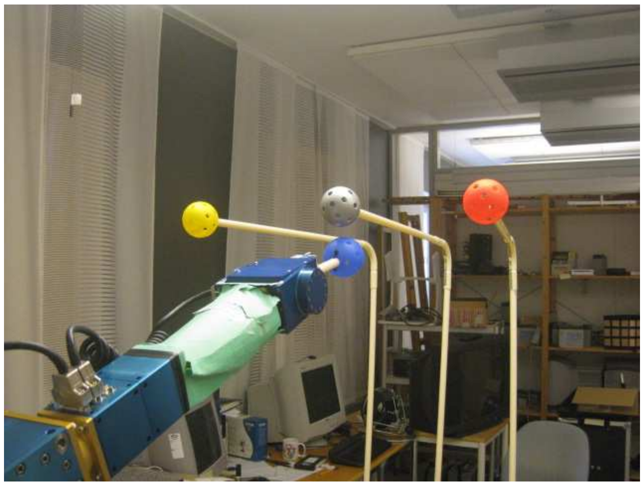


Fig. 6. The manipulator and targets. From left to right, the target colors are yellow, gray, and red.

1) Aid systems: In the user experiment, two types of aid systems were also implemented and evaluated. These systems grant the user less freedom of motion, but give higher precision.

The first aid system used virtual fixtures. The resulting motion from the MJ based system is projected onto a straight line from the starting point to each of the possible goal points. The line with the shortest euclidian distance to the estimated position is chosen. Using this approach, the possible motion trajectories of the robot arm are along a straight line towards one of the three goal points, and the user can control which goal to move towards, when and how fast to do this, and how far towards the goal to move.

The second aid system implemented task control, for the highest possible precision. In this control mode, the user's input is used to choose between 4 possible predefined tasks:

1 Touch red target
2 Touch gray target
3 Touch yellow target
4 Stay in start position

In order to choose which task to execute, a polynomial trajectory estimate is calculated as in the MJ control system, and the endpoint of the motion is extrapolated. The target closest to the endpoint is then touched. If the endpoint of the motion is not close enough (within 15 cm) to any of the targets, task 4 is chosen and the arm remains motionless. In the first few centimeters of a motion, before an accurate target estimate is available, the arm will follow the directly tracked position of the wiimote. The arm will therefore start moving almost at the same time as the user, long before the

TABLE II

THE PERCENTAGE OF HITS WITH EACH OF THE THREE SYSTEMS.

	MJ Control	Virtual Fixtures	Task Control
median	37.5%	72.5%	85%
best	75%	97.5%	100%
worst	17.5%	37.5%	42.5%

user's motion is completed, and if the user moves straight towards one of the possible target postions, the result will be an exact mimic of the user's motion in real time. If the extrapolated endpoint is recalculated to be closer to another target during the execution of a motion, the task will be changed in mid-motion.

C. Task

Each subject was asked to stand next to the robot so that all motions were carried out in parallel, as shown in Figure 7. The colors "red", "gray" and "yellow" were called out to the subject, who in turn tried to touch the corresponding target with the robot end effector by carrying out the same motion with the wiimote.

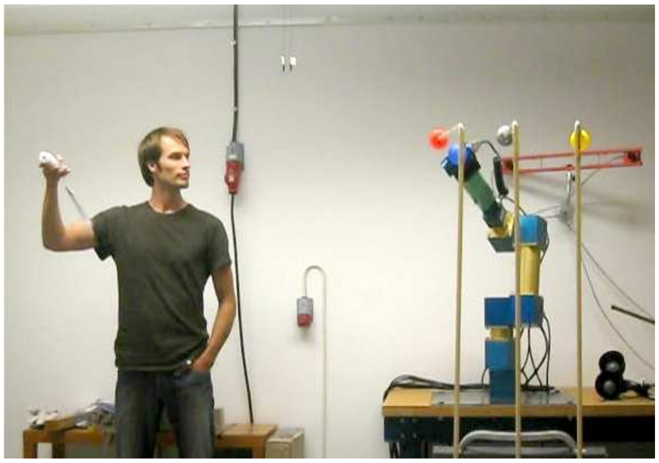


Fig. 7. The subject and the robot.

The subjects were asked to first perform the 40 motions with the MJ based tracker directly controlling the robot in real time, and then repeat the procedure with first the virtual fixtures and then the task control system. Each subject had to touch the 40 targets in a predetermined order that was identical for all experiments. In total, 120 motions were recorded for each subject. The fastest subjects took approximately 12 minutes to complete the experiment, the slowest subject needed 18 minutes.

D. Performance

The task completion performance was measured in a "hit-or-miss" fashion. Each time the user would hit the indicated target in one try without hitting any other targets counted as a hit, and a failure to do this, however close, counted as a miss. This means that the allowed margin of error is the diameter of the balls, 6 cm. The results are given in Table II.

E. Trajectory Estimation Performance

While the "hit-or-miss" metric gives an overview of the performance, analyzing the tracked positions of the wiimote controller gives quantitative measure of precision. Figure 8 shows the cumulative percentage of tries that came closer than given distance of the targets. The most eye-catching result is that the performance is the worst for the yellow target, which had the right-handed subjects reaching to their left. Another observation is that for the most part, the user input has higher precision for the MJ mode. This is to be expected, as the subjects are likely to notice that they can complete the tasks with less effort when different levels of aids are added.

The motion caption system we employ is only usable within the robot workspace, and it was therefore not possible to use this for ground truth measurements when users were controlling the robot. We assume, however, that the precision of the MJ based tracker in this experiment is similar to what was shown for the experiments in Section IV-B. If we isolate the three best performing users, we see that their average errors — as measured as the distance from the actual target position — are only slightly larger than the inherent error in the tracker. Their average error was 11 cm.

The average error over all 15 subjects was 19 cm in total, but the error in the horizontal plane was smaller than the error in the vertical direction, 11 cm as opposed to 16 cm. A probable reason for this is that moving too far in the vertical direction would still score a "hit", while a miss in a horizontal direction resulted in a "miss".

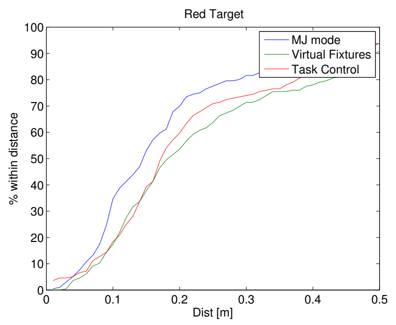
F. Observations

Some subjects were fairly close to the targets when they missed in MJ control mode. The performance of these subjects improved greatly from the added aid of virtual fixtures or task control, the largest improvement being from 22.5% success without aids to 87.5% and 92.5% success rates for the two aided modes.

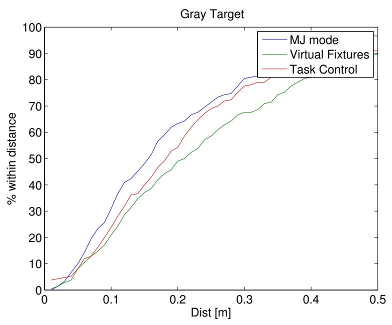
Other subjects, with a larger standard deviation in their input, did not improve their performance with added aids. Indeed, some subjects' performance even degraded. They made motions that were too large and rapid, in effect saturating the acceleration sensors. They thus miss the targets by more than 15 cm. The effect of this is that the robot does not move as they expect it too. They make the false assumption that the motion was too small for the control system to register, and adjust by making even more exaggerated motions, further worsening the performance.

VI. CONCLUSIONS

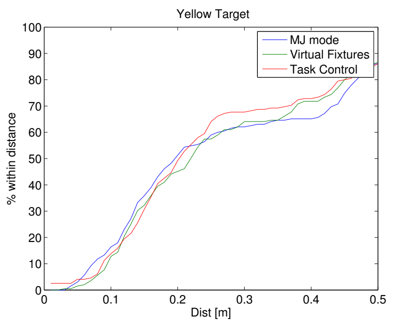
When estimating hand trajectories from the accelerometer data gathered from a wiimote, imposing minimum jerk motion models improves performance significantly over simply integrating the acceleration or using Kalman filtering. One of the main advantages with the MJ approach is that drift caused by buildup error is eliminated, but the MJ based tracker performs better even when the buildup error is manually removed from the Kalman tracker.



(a) Cumulative performance for red target.



(b) Cumulative performance for gray target.



(c) Cumulative performance for yellow target.

Fig. 8. User performance

However, the precision is still only good enough for the simplest of manipulation tasks, and when physically separated from the task, as in the robot experiment described in Section V, the users themselves introduce a larger positioning error than the input tracking system.

Adding virtual fixtures greatly improves precision while retaining enough sense of direct control to allow users to improve their input motions. Even the completely novice users in the experiment in the present paper could achieve good success rates after a few minutes of operation. Giving more control to the robot, as in task control mode, raises the success rates further for most users. However, this control mode can be less intuitive and users that have poor performance may not understand how to improve it.

Given the results in the present paper, it is reasonable to expect future applications utilizing these types of intuitive controllers for manipulation tasks. Several simple modifications could also be added to the approach described here in order to raise precision. As suggested for other applications, a tetrahedronal IR array can be used to give absolute 6 DOF pose measurements when the wiimote is centered on the array [19]. This could be combined with the approach used in the present paper to cover a large range of possible poses. Also, it is highly probable that the Wii MotionPlus, a gyroscope sensor extension for the wiimote, can be used to further improve motion tracking.

Furthermore, as more and more handheld devices such as cell phones and PDAs are equipped with accelerometers and other sensors, this approach could also be applied to novel input interfaces for these.

References

[1] C. Guo and E. Sharlin, "Exploring the use of tangible user interfaces for human-robot interaction: A comparative study," in *Proceeding of the SIGCHI Conference on Human Factors in Computing Systems*, 2008, pp. 121–130.

[2] M. Lapping-Carr, O. Jenkins, D. Grollman, J. Schwertfeger, and T. Hinkle, "Wiimote interfaces for lifelong robot learning," in *AAAI Symposium on Using AI to Motivate Greater Participation in Computer Science*, Palo Alto, CA, USA, Mar 2008. [Online]. Available: http://www.cs.brown.edu/~cjenkins/papers/SS08LappingCarrM_aaai2008.pdf

[3] A. Gams and P.-A. Mudry, "Gaming controllers for research robots: controlling a humanoid robot using a WIIMOTE," in *17th International Electrotechnical and Computer Science Conference (ERK08)*, 2008.

[4] P. Varcholik, D. Barber, and D. Nicholson, "Interactions and training with unmanned systems and the nintendo wiimote," in *The Interservice/Industry Training, Simulation, and Education Conference*, Orlando, Fl., Dec 2008.

[5] P. Marks, "Wii and iPhone help military control freaks," *New Scientist*, p. 26, Mar 29 2008.

[6] T. Shiratori and J. K. Hodgins, "Accelerometer-based user interfaces for the control of a physically simulated character," *ACM Trans. Graph.*, vol. 27, no. 5, pp. 1–9, 2008.

[7] T. H. Song, J. H. Park, S. M. Chung, S. H. Hong, K. H. Kwon, S. Lee, and J. W. Jeon, "A study on usability of human-robot interaction using a mobile computer and a human interface device," in *Intl. Conf. on Human Computer Interaction with Mobile Devices and Services (Mobile HCI'07)*. ACM, Sep 2007, pp. 462–466.

[8] C. Connolly, "Kuka robotics open architecture allows wireless control," *Industrial Robot: An International Journal*, vol. 31, no. 1, pp. 12–15, 2008.

[9] B. Barshan and H. F. Durrant-Whyte, "An inertial navigation system for a mobile robot," in *Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Jul 1993, pp. 2243–2248.

[10] H. H. S. Liu and G. K. H. Pang, "Accelerometer for mobile robot positioning," *IEEE Transactions on Industry Applications*, vol. 37, no. 3, pp. 812–819, May/Jun 2001.

[11] J. Kela, P. Korpipää, J. Mäntyjärvi, S. Kallio, G. Savino, L. Jozzo, and D. Marca, "Accelerometer-based gesture control for a design environment," *Personal Ubiquitous Comput.*, vol. 10, no. 5, pp. 285–299, 2006.

[12] N. Hogan, "An organizing principle for a class of volontary movements," *Journal of Neuroscience*, vol. 4, pp. 2745–2754, 1985.

[13] T. Flash and N. Hogan, "The coordination of arm movements: An experimentally confirmed mathematical model," *The Journal of Neuroscience*, vol. 5, no. 7, pp. 1688–1703, July 1985.

[14] M. Kawato, "Trajectory formation in arm movements: Minimization principles and procedures," in *Advances in Motor Learning and Control*, ser. Human Kinetics, H. N. Zelaznik, Ed., 1996, pp. 225–259.

[15] A. Hauck, M. Sorg, and T. Schenk, "What can be learned from human reach-to-grasp movements for the design of robotic hand-eye systems?" in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 4, Detroit, Michigan, USA, 1999, pp. 2521–2526.

[16] C. Smith, M. Bratt, and H. I. Christensen, "Teleoperation for a ballcatching task with significant dynamics," *Neural Networks, Special Issue on Robotics and Neuroscience*, vol. 24, pp. 604–620, May 2008.

[17] M. Bratt, C. Smith, and H. I. Christensen, "Minimum jerk based prediction of user actions for a ball catching task," in *IROS-07*. San Diego, Ca, USA: IEEE/RSJ, Oct 2007.

[18] C. Smith and H. I. Christensen, "Using COTS to construct a high performance robot arm," in *Proceedings of the 2007 IEEE International Conference on Robotics and Automation*. Rome, IT: IEEE, April 2007, pp. 4056–4063.

[19] O. Kreylos, "Oliver kreylos' research and development homepage - wiimote hacking," Accessed Sep 15, 2008. [Online]. Available: http://idav.ucdavis.edu/~okreylos/ResDev/Wiimote/index.html