PALADYN Journal of Behavioral Robotics

Research Article · DOI: 10.2478/pjbr-2013-0017 JBR · 4(3) · 2013 · 182-193

Application of the *Novint Falcon* haptic device as an actuator in real-time control

Daniel J. Block[1,*], Mark B. Michelotti[2,†], Ramavarapu S. Sreenivas[3,‡]

1 College of Engineering Control Systems Laboratory, University of Illinois at Urbana-Champaign, Urbana IL 61801, USA

2 Navistar Inc., Barrington, IL 60010, USA

3 Coordinated Science Laboratory & Industrial and Enterprise Systems Engineering, University of Illinois at Urbana-Champaign, Urbana IL 61801, USA

Received 13-10-2013

Accepted 25-11-2013

Abstract

This paper describes the development of an embedded system whose purpose is to control the *Novint Falcon* as a robot, and to develop a control experiment that demonstrates the use the *Novint Falcon* as a robotic actuator. The *Novint Falcon*, which is a PC input device, is "haptic" in the sense that it has a force feedback component. Its relatively low cost compared with other platforms makes it a good candidate for academic application in robot modeling and control. An embedded system is developed to interface with the multiple motors and sensors present in the *Novint Falcon*, which is subsequently used to control three independent *Novint Falcons* for a "ball-on-plate" experiment. The results show that the device is a viable solution for high-speed actuation of small-scale mechanical systems.

Keywords

haptic device · embedded systems · electromechanical systems

1. Introduction

The *Novint Falcon* is a haptic device (cf. figure 1) that is intended to serve as a replacement for a joystick or mouse. In its structure and design it is a small robot that permits the user to experience virtual touch in a seamless manner. The user moves the spherical grip around in 3D space, providing input to a PC.

Although it is intended to be used as a type of force-feedback PC game controller, other applications have been explored, including robotics [2]. The relatively low cost of the Novint Falcon makes it a good candidate for a high-volume academic robotic platform. Actuation can be accomplished by using the force-feedback motors to move the manipulator. Robotic control of the Novint Falcon requires a kinematic understanding of the mechanical configuration of the device, as well as a digital computer to perform necessary control calculations. The device's mechanical system remains unchanged for the purposes outlined in this paper; however, its electrical system is bypassed to allow direct control of the force-feedback motors.

The use of alternate approaches to the design of PI, PD and PID controllers have received attention from several researchers. The classical design methods used in this paper could be augmented and improved by the use of these newer approaches [13–17].

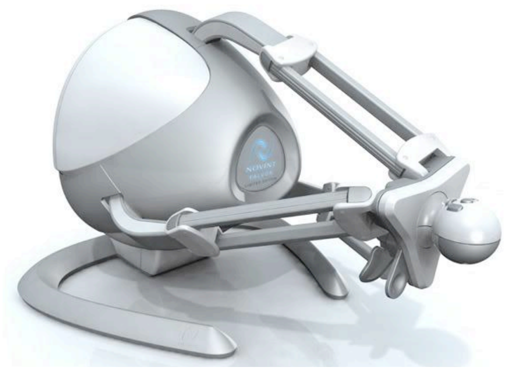


Figure 1. The Novint Falcon 3D controller.

1.1. Kinematic Configuration

The mechanical configuration of the Novint Falcon was first introduced by Tsai and Stamper in their 1997 technical research report [4]. The

*E-mail: d-block@illinois.edu
†E-mail: mark.michelotti@gmail.com
‡E-mail: rsree@illinois.edu

Novint Falcon's configuration is identical to that presented in this report, which has several attractive characteristics.

- The kinematics have closed-form solutions.
- Position and orientation of the manipulator are uncoupled.
- The construction uses only revolute joints, resulting in a lower hardware cost.

The Novint Falcon consists of a stationary platform and a moving platform. In general, the stationary platform is attached to the world coordinate frame, and the moving platform can be thought of as the manipulator. The two platforms are connected by three identical parallel kinematic chains, much like the common delta-configuration robot. A simplified schematic of figure 1 is shown in figure 2. It differs from figure 1 in that the device is facing upward instead of sideways. The links are labeled by numbers 0 through 16, where the stationary platform is labeled 0 and the moving platform is labeled 16.
There are four links in each of the three kinematic chains. The first link in each chain (links 1, 2, and 3) is connected to the stationary platform, equally spaced from the other two lowest links. The next four links form a parallelogram connected to the moving platform. The four-bar parallelogram consists of links (4, 7, 10, 13) for the first chain, (5, 8, 11, 14) for the second chain, and (6, 9, 12, 15) for the third chain. The three parallelograms are connected to the moving platform with equal spacing.
The result of this configuration is a moving platform with only translational degrees of freedom - that is, the moving platform will always have the same orientation, but its position in 3D space can be controlled by actuating only the lowest three joints. Accordingly, the Novint Falcon has sensors to detect the angular position of only the lowest three joints. An analysis of the constrained degrees of freedom of the device can be found in the paper by Tsai and Stamper [3], and is presented here for the sake of completeness.

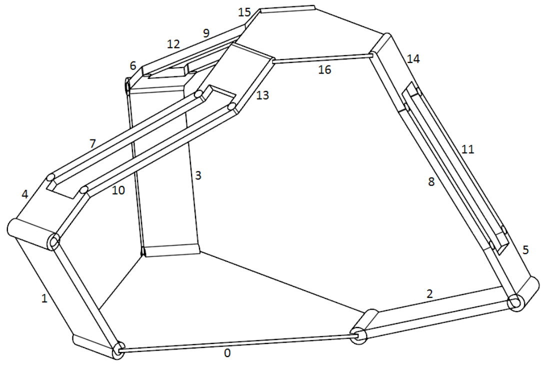


Figure 2. Mechanical schematic of the Novint Falcon's robotic configuration.

1.2. Inverse Kinematics [4]

The inverse kinematics problem for this platform can now be stated: Given the (x, y, z) position of the center of the moving platform, find the angular position of the lowest three joints. Figure 3 is a side view schematic of one of the three kinematic chains. The subscript i denotes the i-th kinematic chain, where i is 1, 2, or 3.

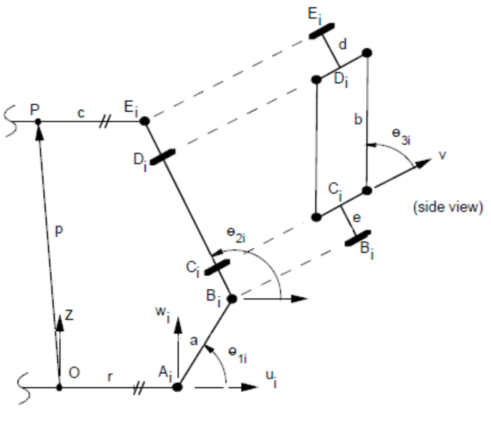


Figure 3. Side view of the i-th kinematic chain (Figure 2, [4]).

The coordinate frame (x, y, z) is attached to the center of the stationary platform. The origin (the center of the stationary platform) is labeled point O, and the center of the moving platform is labeled point P, p is the vector from the center of the stationary platform to the center of the moving platform. The distance from the center of the stationary platform to the lowest joint (joint A_i) is denoted r, and the distance from the center of the moving platform to the highest joint (joint E_i) is denoted c. The lengths of the links are labeled a, b, d, and e. The angular positions of links A_i, B_i, and C_i are given by Θ_{1i}, Θ_{2i}, and Θ_{3i}, respectively. A coordinate frame (u_i, v_i, w_i) is defined for each kinematic chain, attached at point A_i.
The following coordinate transformation expresses the position of point P in the (u_i, v_i, w_i) coordinate system.

$$p_i = \begin{pmatrix} \cos\phi_i & \sin\phi_i & 0 \\ -\sin\phi_i & \cos\phi_i & 0 \\ 0 & 0 & 1 \end{pmatrix} p + \begin{pmatrix} -r \\ 0 \\ 0 \end{pmatrix}$$

where ϕ_i is the angle of rotation of the i-th kinematic chain with respect to the (x, y, z) coordinate frame, and $p_i = (p_{ui}\, p_{vi}\, p_{wi})^T$. Expressions for p_{ui}, p_{vi}, and p_{wi} are given by:

$$\begin{aligned} p_{ui} &= a\cos\theta_{1i} - c + (d + e + b\sin\theta_{3i})\cos\theta_{2i} \\ p_{vi} &= b\cos\theta_{3i} \\ p_{wi} &= a\sin\theta_{1i} + (d + e + b\sin\theta_{3i})\sin\theta_{2i} \end{aligned}$$

The solution for θ_{3i} is

$$\theta_{3i} = \pm\cos^{-1}\frac{p_{vi}}{b}.$$

With θ_{3i} known, an equation with θ_{2i} as the only unknown is generated by isolating the θ_{2i} terms in the equations for p_{ui}, and p_{wi} and then summing the squares of those two equations so that θ_{2i} is eliminated with the application of the Pythagorean relationship as follows

$$(p_{ui} + c)^2 + p_{wi}^2 + a^2 - 2a(p_{ui} + c)\cos\theta_{1i} - 2\alpha p_{wi}\sin\theta_{1i} = (d + e)^2 + 2(d + e)b\sin\theta_{3i} + b^2\sin{\theta_{3i}}^2. \quad (1)$$

Letting $t_{1i} = \tan\frac{\theta_{1i}}{2}$, we have

$$\sin\theta_{1i} = \frac{2t_{1i}}{1+t_{1i}^2} \text{ and } \cos\theta_{1i} = \frac{1-t_{1i}^2}{1+t_{1i}^2},$$

substituting the above expressions into equation 1 and simplifying, the following equation is obtained

$$l_{2i}t_{1i}^2 + l_{1i}t_{1i} + l_{0i} = 0,$$

where

$$\begin{aligned}
l_{0i} &= p_{wi}^2 + p_{ui}^2 + 2cp_{ui} - 2ap_{ui} - b^2\sin^2\theta_{3i} - \\
&\quad 2be\sin\theta_{3i} - 2bd\theta_{3i} - 2de - 2ac + a^2 + c^2 - d^2 - e^2 \\
l_{1i} &= -4\alpha p_{wi} \\
l_{2i} &= p_{wi}^2 + p_{ui}^2 + 2cp_{ui} - 2ap_{ui} - b^2\sin^2\theta_{3i} - \\
&\quad 2be\sin\theta_{3i} - 2bd\theta_{3i} - 2de - 2ac + a^2 + c^2 - d^2 - e^2
\end{aligned}$$

This quadratic equation can be solved for t_{1i}, which gives two possible values for θ_{1i} for each of the two possible values of θ_{3i}. θ_{2i} can then be found by substituting these values into the initial expression of the manipulator position. Thus, there are four possible solutions for any given (x, y, z) position. In the configuration of the Novint Falcon, the range of motion of joint C prevents angle θ_{3i} from being negative. This eliminates two solutions. Similarly, θ_{1i} is always in the first quadrant, which eliminates the remaining ambiguity, leaving one solution [3].

1.3. Forward Kinematics

The forward kinematics problem is: Given $(\theta_{11}, \theta_{12}, \theta_{13})$, find the (x, y, z) position of the center of the moving platform. However, the closed-form analytical solution to this problem is significantly more complex. Tsai and Stamper [3] have shown that the parallel configuration of the manipulator results in 16 forward kinematic solutions for any given set of values for the angular positions of the lowest three joints. The solution involves solving a high-degree polynomial [2, 3]. Since the forward kinematics function is expected to be executed every sampling period (1 kHz), it is desired to avoid this complexity. Nonetheless, if closed-loop position control is to be achieved, a computation of the forward kinematics is required. Fortunately, there are other methods available for this computation. One alternative is to use a look-up table and interpolation. The main advantage with a look-up table is that it needs to be generated only once, and subsequent use involves computationally simple interpolation. The disadvantage is that some accuracy is sacrificed, depending on the resolution of the table. To achieve higher accuracy, we recognize that the forward kinematic problem is a system of nonlinear equations in three variables. Therefore, a nonlinear zero-finding method such as Broyden's method can be used [5]. Since this is an iterative method, it is not guaranteed to converge to the true solution unless the "starting guess" is relatively close to the true solution. Given this limitation, a combination of a look-up table and Broyden's method is used. The look-up table provides an approximate solution, which is then used as a starting point for Broyden's method, which converges to the true solution. In a real-time control environment, it is assumed that this computation will occur every sampling period. In light of this, another good starting point for Broyden's method is simply the position of the manipulator at the previous time step. Using these heuristics, Broyden's method converges to less than 0.1 mm error in 1-2 iterations.

1.4. Implementation of Kinematics

The relevant dimensions of the Novint Falcon are enumerated in figure 4. Computation of the inverse kinematics for the first kinematic chain is implemented using the code shown in figure 5. The implementation of Broyden's method to solve the forward kinematics problem uses the inverse kinematics function iteratively. Broyden's method takes the general form of the code shown in figure 6.

Dimension	Value
a	60 mm
b	103 mm
c	16.3 mm
d	12 mm
e	12 mm
r	37 mm
φ_1	-14.44°
φ_2	-105.56°
φ_3	-225.56°

Figure 4. Dimensions of the Novint Falcon.

```
function [theta1] = inverse_kinematics(x, y, z)
r = 37;
a = 60;
b = 103;
c = 16.3;
d = 12;
e = 12;
phi = -0.252 %radians

pu = x*cos(phi) + y*sin(phi) - r;
pv = -x*sin(phi) + y*cos(phi);
pw = z;

theta3 = acos(pv/b);

l0 = pw^2+pu^2+2*c*pu-2*a*pu+a^2+c^2-d^2-e^2-b^2*sin(theta3)^2-
2*b*e*sin(theta3)-2*b*d*sin(theta3)-2*d*e-2*a*c;
l1 = -4*a*pw;
l2 = pw^2+pu^2+2*c*pu+2*a*pu+a^2+c^2-d^2-e^2-b^2*sin(theta3)^2-
2*b*e*sin(theta3)-2*b*d*sin(theta3)-2*d*e+2*a*c;

t1 = (-l1-sqrt(l1^2-4*l2*l0))/(2*l2);
theta1 = atan(t1)*2;
```

Figure 5. Inverse Kinematics (MATLAB).

For the application-specific forward kinematics, $x_k = (x_k\ y_k\ z_k)^T$ is the k-th approximation of the position of the moving platform. f is the inverse kinematics function whose input is the (x, y, z) position of the moving platform and whose output is the three angles of the lowest three joints, $(\theta_{11}\ \theta_{12}\ \theta_{13})^T$. B_k is the k-th approximation of the Jacobian matrix of the nonlinear function f. Note that the derivative of f is not explicitly evaluated. Rather, the Jacobian matrix is successively

```
$x_0$ = initial guess
$B_0$ = initial Jacobian approximation
for k = 0, 1, 2, ...
        Solve $B_k s_k = -f(x_k)$ for $s_k$
        $x_{k+1} = x_k + s_k$
        $y_k = f(x_{k+1}) - f(x_k)$
        $B_{k+1} = B_k + ((y_k - B_k s_k) s_k^T)/(s_k^T s_k)$
end
```

Figure 6. Broyden's Method (pseudocode).

approximated. Since the Jacobian of the inverse kinematics function is difficult to compute, this is a desired property [6].

2. Control

The required components for feedback control of the Novint Falcon are available in the device. Three DC motors actuate the lowest three joints, and three encoders provide position feedback for the lowest three joints. To achieve control of the (x, y, z) position of the manipulator, the Novint Falcon can be considered to have three inputs and three outputs. The three inputs are the voltages across the three DC motors that drive the lowest joint of each kinematic chain. The three outputs are the x, y, and z position of the center of the moving platform. Using the kinematics derived in the previous section to provide desired motor angular positions, the system can be posed as three single-input single-output (SISO) systems, rather than one multiple-input multiple-output (MIMO) system. The angular position of each motor will then be uncoupled and controlled individually. Position control of each DC motor is attained using the common PID controller by selecting appropriate gains. A block diagram for feedback control of a single DC motor using this classical control strategy is shown in figure 7.

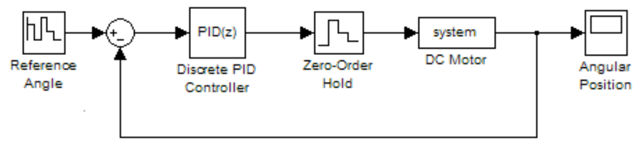


Figure 7. PID control strategy for a DC motor.

It is difficult to compute gains analytically since the plant characteristics of the DC motor are unknown and nonlinear. The nonlinearities are due to the nonlinear dynamics of the device. Therefore, an alternative manual approach is taken. The gains can be tuned experimentally until a suitable response is achieved. Once the appropriate gains have been found, the position of the manipulator can be controlled to a point in its workspace. A block diagram for this manipulator position control strategy is shown in 8. In this block diagram, the "Motor Control Loop" blocks represent the system of the Figure 7 for each motor. The loop is closed in these blocks rather than in an "outer loop" that feeds back the (x, y, z) position. This is done for two reasons. First, to directly feed the sensor output back, rather than an output value of the nonlinear forward kinematics function. Second, to avoid having to input an error term rather than an absolute position into the inverse kinematics function. It is easy to see that the behavior of the position error of the manipulator is directly related to the behavior of the motor angular error by the inverse kinematic function. The conclusion is that a well-designed motor control will result in a well-behaved position control; e.g., the step response of the moving platform will have the same time constant as the step response of the DC motors. So, the goal is to design the controller of figure 8 to satisfy some design specifications. The manual control gain tuning procedure is enumerated here:

1. Find some intuitive starting point for k_p, and set k_i and k_d to zero.
2. Tune k_p until the response speed is fast. There may be some overshoot in the step response.
3. Increase k_d until the overshoot decreases to an acceptable level.
4. Repeat steps 2 and 3 until the response is as fast as possible with acceptable overshoot.
5. Determine the steady-state error due to static friction and increase k_i accordingly.

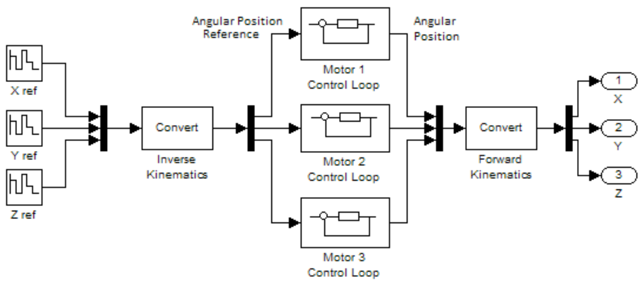


Figure 8. Position control strategy for the Novint Falcon.

It is important to note that the goal is to design a feedback control such that the motor position will track a continuous trajectory with high fidelity. Although the design is largely taking place with respect to step response specifications, a good step response will result in good trajectory tracking. After tuning, the gains resulting in the best response are $k_p = 20$, $k_d = 0.2$, and $k_i = 1$. The step response of the Novint Falcon moving from $(x, y, z) = (-22, 14, 135)$ to $(x, y, z) = (-4, 3, 112)$ is shown in figure 9, as well as the response of the motors. Figure 10 shows the performance when tracking a sinusoidal trajectory input. In both figures, the dotted line is the reference input and the solid line is the output.

As a final note, in all results presented in this section, difference equations are computed from discrete representations of the designed controller and implemented on a digital computer. A discrete differentiator has the form $D(z) = \frac{z-1}{T_s z}$, and a discrete integrator is of the form $D(z) = \frac{T_s}{z-1}$, where $T_s = 10^{-3}$ seconds is the sampling period of the discrete system.

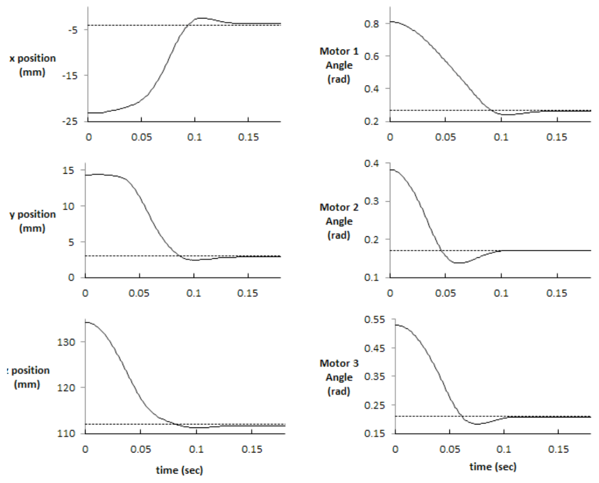


Figure 9. Step response of the designed position controller.

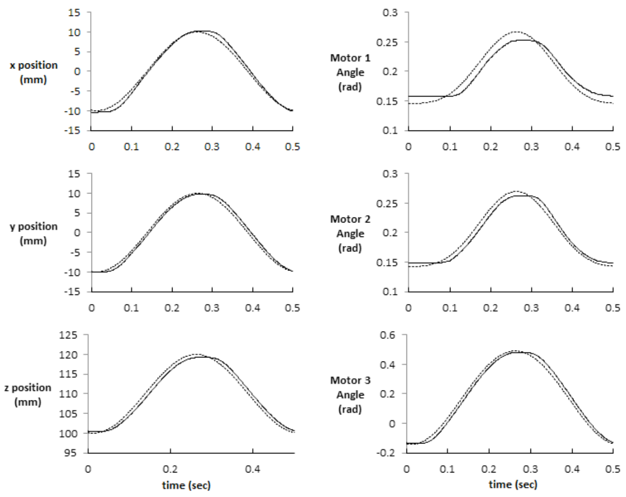


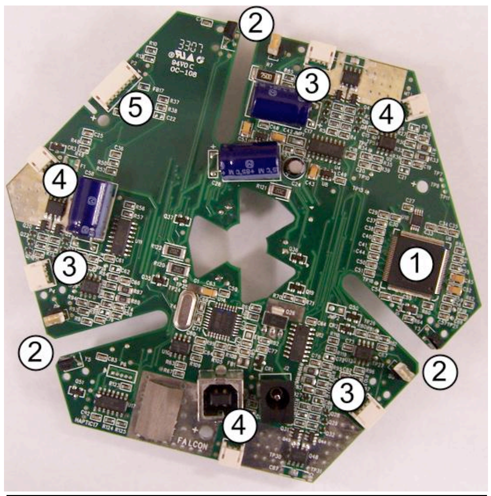
Figure 10. Trajectory response of the designed position controller.

3. Hardware Implementation

The Novint Falcon device communicates to a controlling computer through a USB (Universal Serial Bus) port. Novint reports a sampling rate of 1 kHz through the USB interface. However, Martin and Hillier [2] reported that this update rate was not sustained with high fidelity, and typically missed commands or reads resulted in a real-world communication rate between 800 Hz and 1 kHz, depending on the controlling computer's load. In other words, the non-realtime nature of a PC operating system contributed to variations in the sample rate. The PC interface also resulted in a 2-5 sample delay between commands being issued and results being received. It is desired to control the three motors in the Novint Falcon at a hard real-time sampling rate of 1 kHz. This type of high-fidelity control can be achieved with an embedded processor running a real-time operating system. There are several challenges in implementing this hardware strategy, including:

1. Overriding the existing embedded system in the Novint Falcon
2. Choosing a suitable processor
3. Driving the three DC motors
4. Reading the three motor position sensors

To override the existing embedded system in the Novint Falcon, the circuit board in the device must be analyzed. It is shown in figure 11. The controlling processor is indicated in figure 11. Removing this chip results in a more "static" system; i.e., individual components of the Novint Falcon are not enabled and disabled unexpectedly. It is important to note that after this modification, the device will be permanently disabled for its intended purpose as a haptic input controller.



Component	Description
1	TMS320 Digital Controller
2	Encoder LED emitters and photosensors
3	Supplementary Sensors
4	Motor Leads
5	Controller Buttons

Figure 11. The embedded circuit board in the Novint Falcon, and a description of its components.

3.1. Embedded Processor

The basic requirements for the embedded processor to control the Novint Falcon are enumerated here.

1. Powerful enough to handle a 1 kHz sample rate
2. Capable of driving a DC motor (pulse-width modulated outputs or DAC outputs)
3. Sensor inputs to handle motor position feedback and supplementary sensors

The Texas Instruments TMS320F28335 Delfino Microcontroller controlCard is a complete board-level module in an industry-standard Dual In-line Memory Module (DIMM) form factor, shown in figure 12 [7]. The F28335 satisfies all the basic requirements for control of the Novint Falcon –

1. It has a 150 MHz clock speed and is more than capable of a 1kHz sample rate. It also has a floating-point unit, making control calculations more efficient.

2. Its multiple pulse-width modulated outputs can be amplified to drive the three motors of the Novint Falcon

3. It has quadrature counters to interface with the encoders attached to each motor, as well as a Serial Peripheral Interface (SPI) to communicate with external quadrature counter chips. Its many General Purpose Input/Output (GPIO) pins can serve as inputs for any supplementary sensors.

Figure 12. The F28335 controlCard [7].

3.2. Driving the DC Motors

The F28335 is capable of producing a pulse-width modulated (PWM) signal with a varying duty cycle with an amplitude of 3.3 V [7]. In order to drive one of the DC motors in the Novint Falcon, this signal needs to be amplified to ±12 V. This way, a 50% duty cycle will result in zero motor torque, 100% will be full torque in one direction, and 0% will be full torque in the opposite direction. This signal amplification is achieved with the A3953 Full-Bridge PWM Motor Driver chip from Allegro MicroSystems, Inc [8]. A schematic of this chip is shown in figure 13. For this application, the pulse-width modulated signal will be input at the PHASE pin (pin 7). OUT_A will be connected to the motor's negative terminal, and OUT_B will be connected to the motor's positive terminal. The motor terminals are indicated in figure 11 by blue squares. The $LOAD\ SUPPLY$ pins should be connected to +12V. The $BRAKE$ pin will be pulled up to 3.3V to disable that function.

3.3. Angular Position Feedback

Each of the three DC motors in the Novint Falcon is equipped with a large encoder wheel along with a light-emitting diode (LED) and photosensor. These are indicated in figure 11 by green squares. As the encoder wheel turns along with the motor, the gaps in the wheel pass between the LED and photosensor, generating quadrature encoder signals. Keeping a count of these signals provides an accurate measure

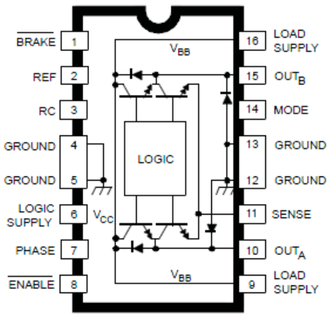


Figure 13. Schematic of the A3953 Full-Bridge PWM Motor Driver [8].

of how far and in which direction the motor has turned. This is accomplished with the LS7366R Quadrature Counter with Serial Peripheral Interface (SPI). A schematic of this chip is shown in figure 14 [9]. The previously mentioned SPI protocol is fully compatible with the F28335 [8]. The quadrature encoder channels A and B are connected to the A and B pins, respectively (pins 12 and 11). The SPI bus is serviced on the SS, SCK, $MISO$, and $MOSI$ pins (pins 4, 5, 6, and 7).

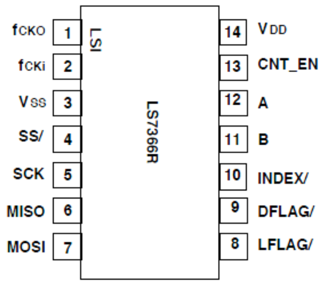


Figure 14. Schematic of the LS7366R Quadrature Counter [9].

3.3.1. Supplementary Sensors

Each of the three DC motors in the Novint Falcon is equipped with a "home position" sensor that is tripped when the motor is at a specific angle. The sensor leads are indicated in figure 11 by pick squares. These sensor outputs can be connected to GPIO inputs on the F28335 to detect the absolute angular position of the motors.

4. The Ball-on-Plate System

We now describe the design of a laboratory control experiment that uses the Novint Falcon as an actuator, proving the concept of the device as a robotic manipulator. The mechanical system to be controlled is the "ball-on-plate" system, which consists of a ball free to roll around on a flat plate. The applied control would presumably "balance" the ball at a certain position on the plate, or control the position of the ball on the plate [5] [6] [7]. This system was chosen for several reasons.

1. In its chosen implementation, the experiment requires the use of three cooperating Novint Falcon devices. This will demon-

strate the capabilities of the embedded approach to the control system.

2. This experiment will involve some vision processing for detecting ball position. Visual servo control can then be demonstrated in conjunction with the Novint Falcon.

3. The system has a slow response; i.e., the ball rolls slowly enough that the effects of variations in control parameters can be easily observed, and the Novint Falcon's response time is significantly faster than the control experiment.

The chosen implementation for the ball-and-plate system is shown in a schematic in figure 15. Three Novint Falcon devices are arranged facing upwards and equally spaced. The plate rests on top of the three moving platforms. This way, the position of the moving platforms determines the tilt angle of the plate.

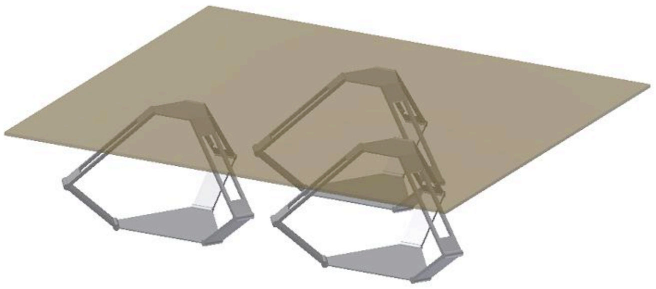


Figure 15. Schematic of the ball-and-plate system implementation.

5. Ball Position Feedback

Ball position feedback is accomplished with a camera facing downwards from directly above the plate [10]. A wide-angle lens is used to maximize the viewing angle of the camera. Vision processing algorithms are employed to threshold and segment the image in order to find the ball's position [12]. The vision processing (along with the control algorithm) is implemented on the Digital Signal Processor (DSP) in an OMAP-L138 SoC system [11]. It was observed that the distortion in the wide-angle lens used is predominantly radial in nature. This "fisheye" distortion is illustrated in figure 16.

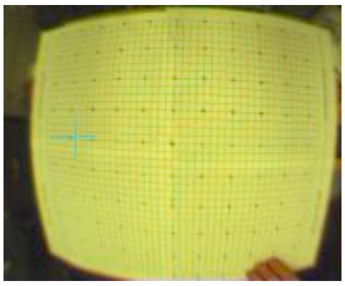


Figure 16. Observed wide-angle lens "fisheye" distortion.

A mapping from pixel position to actual position is desired. With the assumption that all of the lens distortion is radial, all that remains is to develop a function g(r) that maps a pixel's distance from the center pixel to the real distance of that point from the center of the plate. This is accomplished by imaging a square grid and compiling a table of pixel distances and true distances, then fitting a least-squares polynomial to the data. The data and a least-squares quadratic polynomial is shown in figure 17.

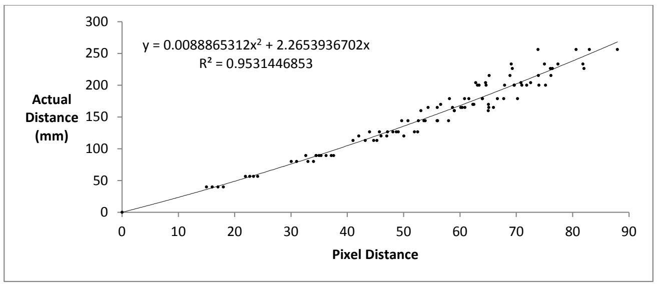


Figure 17. Results of lens distortion analysis.

6. Dynamic Model

As mentioned before, the mechanical system to be modeled is the "ball-on-plate" system, which consists of a ball free to roll around on a flat plate. The applied control would presumably "balance" the ball at a certain position on the plate, or control the position of the ball on the plate. This is accomplished by changing the angle of the plate. A simple schematic of he system is shown for the one-dimensional "ball-on-beam" system in figure 18. A hollow ball is used for this analysis. It is also assumed that the ball-on-plate system is a two-dimensional analog of the ball-on-beam system. It is further assumed that the pivot point of the beam is at the point of contact between the ball and beam. Thus, there are no lever-arm forces applied to the ball when the angle changes. The dynamics for this simplified system can be found using the torque equation, $\tau = I\alpha$. Since $\frac{d^2}{dt^2}x = r\alpha$ and $I = \frac{2mr^2}{3}$ the equation of motion is $\frac{d^2}{dt^2}x = \frac{3g}{2}\sin\theta$, which is subsequently approximated (for small θ) as $\frac{d^2}{dt^2}x = \frac{3g}{2}\theta$. The goal of this analysis is to determinea feedback control that regulates the system to $x = 0$. This can be accomplished with both linear and nonlinear feedback.

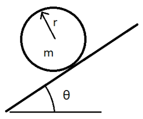


Figure 18. Simplified ball-on-beam system.

6.1. An Implementation that Approximates the Dynamic Model

It is clear from figure 15 that the plate has all six degrees of freedom (in a certain workspace). This freedom allows the plate to pivot about any point within its workspace. We will constrain the plate to pivot about the point of contact between the plate and the ball, approximating the dynamic system described above. The problem statement is this: given two control inputs Θ_x and Θ_y, at what positions must the three moving platforms be to actuate these inputs? The geometric derivation of this process uses the geometric reference shown in figure 19.

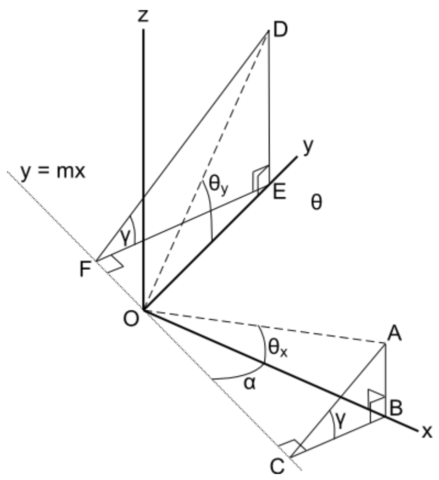


Figure 19. Geometric reference.

First, define a coordinate system (x, y, z) whose origin is the center of the system and at the same height as the ball. Note that in the final implementation, the ball will always be at the same height, so this coordinate frame is static. Triangles OAC and ODE are coplanar. Determine the slope m of the line $y = mx$ and the angle γ as follows: Note that triangles ABC and DEF are similar. So:

$$\frac{\overline{AB}}{\overline{BC}} = \frac{\overline{DE}}{\overline{EF}} \Rightarrow \frac{\sin\theta_x}{\sin\alpha\cos\theta_x} = \frac{\sin\theta_y}{\sin\pi/2-\alpha\cos\theta_y}$$
$$\Rightarrow \frac{\tan\theta_x}{\sin\alpha} = \frac{\tan\theta_y}{\cos\alpha} \Rightarrow m = -\tan\alpha = \frac{-\tan\theta_x}{\tan\theta_y}.$$

The value of angle γ is found using triangel ABC:

$$\tan\gamma = \frac{\overline{AB}}{\overline{BC}} = \frac{\sin\theta_x}{\sin\alpha\cos\theta_x} = \frac{\tan\theta_x}{\sin\alpha}$$

Let the positions of the three moving platforms for $\theta_x = \theta_y = 0$ be denoted p_1, p_2 and p_3. Their perpendicular distance to the line $y = mx$ is given by the expression

$$d_i = \frac{-mp_{ix} + p_{iy}}{\sqrt{m^2+1}}.$$

Consequently, the new x and y positions are

$$\begin{aligned}\tilde{p}_{ix} &= p_{ix} - d_i(1-\cos\gamma)\cos\left(\frac{\pi}{2}+\tan^{-1}m\right)\\ \tilde{p}_{iy} &= p_{iy} - d_i(1-\cos\gamma)\sin\left(\frac{\pi}{2}+\tan^{-1}m\right)\end{aligned}$$

Finally, to fix the ball's height, simply add or subtract the necessary distance from the z component of each of p_1, p_2, and p_3. If the ball's x and y position is given by b_x and b_y, then

$$\tilde{p}_{iz} = \left(d_i - \frac{-mb_x+b_y}{\sqrt{m^2+1}}\right)\sin\gamma$$

All that remains is the trivial task of converting these coordinates to each device's individual frame of reference. Note that with this algorithm, the devices need not be arranged in any specific way.

7. Linear Control Design

The linearized dynamic system can be represented with two states as follows

$$\begin{aligned}\frac{d}{dt}x_1 &= x_2\\ \frac{d}{dt}x_2 &= \frac{3g}{2}\theta.\end{aligned}$$

This linear system is a double integrator with a gain of $3g/2$. Its behavior near the equilibrium is related to the eigenvalues $\lambda_{1,2} = \frac{J_{22}\pm\sqrt{4J_{21}+J_{22}^2}}{2}$, of its Jacobian J, where

$$J = \begin{pmatrix} 0 & 1 \\ \frac{3g}{2}\frac{\partial\theta}{\partial x_1} & \frac{3g}{2}\frac{\partial\theta}{\partial x_2}\end{pmatrix} = \begin{pmatrix} 0 & 1 \\ J_{21} & J_{22}\end{pmatrix}.$$

For asymptotic stability, these eigenvalues must be strictly negative. This is true for any control effort such that

$$\frac{\partial\theta}{\partial x_1} < 0 \text{ and } \frac{\partial\theta}{\partial x_2} < 0$$

The simplest control strategy satisfying these requirements is the common PD control consisting of a linear combination of the position and velocity of the ball: $\theta = -k_p x_1 - k_d x_2$. To track a reference input R, this control law must be modified to act on an error signal as follows: $\theta = -k_p(R - x_1) - k_d(\frac{d}{dt}R - x_2)$. The closed-loop transfer function with this control effort is (cf. figure 20)

$$H(s) = \frac{(3g/2)k_d s + (3g/2)k_p}{s^2 + (3g/2)k_d s + (3g/2)k_p}.$$

Before k_p and k_d are chosen, it is important to note that this linear model is stable for any $k_p > 0$ and $k_d > 0$, and becomes more stable as these gains increase. However, due to the realities of the nonlinear system, it would be catastrophic to choose them to be arbitrarily large. Therefore, as part of the design specification, the step response will have a lower-bounded rise time. The ball will never accelerate faster than 9.81 m/s^2. In fact, with our small angle approximation, we can intuitively say that the ball should never accelerate faster than 1 m/s^2. Coupled with the fact that the step input will be scaled-down, some heuristic design considerations result in a minimum allowable rise time of approximately one second. All that remains is to choose gains k_p and k_d to satisfy some damping consideration to control oscillations, say, $\zeta = 0.6$. So, with $g = 9.81 m/s^2$, we have

$$\begin{aligned} t_r\omega_n &= \frac{1}{\sqrt{1-\zeta^2}}\left(\pi - \tan^{-1}\left(\frac{\sqrt{1-\zeta^2}}{\zeta}\right)\right) = 2.77 \\ &\Rightarrow \omega_n = 2.77 rad/s \\ \omega_n^2 &= 14.751 k_p \Rightarrow k_p = 0.52 \qquad (2) \\ 2\zeta\omega_n &= 14.715 k_d \Rightarrow k_d = 0.23 \qquad (3) \end{aligned}$$

A computer simulation of the step response with these gains and a step of 0.1 is shown in figure 21. The control effort is the dotted line. The next step is to implement this control strategy on the physical system that this model and simulation approximates.

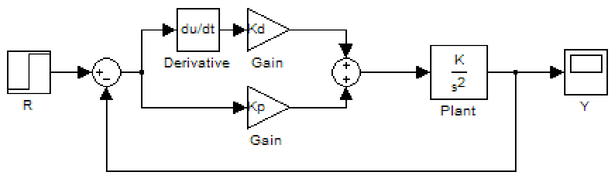


Figure 20. PD control strategy block diagram ($k_p = 0.52$ and $k_d = 0.23$; cf. equations 2 and 3).

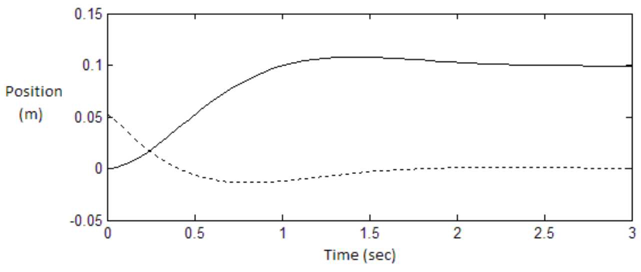


Figure 21. Simulated step response of the PD control (cf. figure 20).

8. Feedback-Linearized Control Design

In contrast to the linear design section, the sinusoidal nonlinearity in the model is considered here. Recall the system equation of motion is $\frac{d^2}{dt^2}x = \frac{3g}{2}\sin\theta$. If we let $\theta = \sin^{-1}(2u/3g)$, the system is described by $\frac{d^2}{dt^2}x = u$. The system can now be mathematically viewed as linear, as long as the control effort u is transformed into θ before input to the plant. Since the plant is now a unity gain double integrator the controller can be designed using linear techniques. A similar analysis to the linear design leads to the same basic control strategy as above – a PD control consisting of a linear combination of the position and velocity of the ball. Using identical specifications, proportional and derivative gains are obtained. In this case, $\omega_n^2 = k_p = 7.67$ and $2\zeta\omega_n = k_d = 3.32$. The simulated step response is similar to the one shown in figure 21 – which is expected, since the small angle approximation used in linear controller design is more or less valid for the simulated instance.

9. Implementation and Results

Controller design was accomplished in the continuous domain. To simulate and implement a PD control on a digital system, the control must be discretized. The control effort will then pass through a zero-order hold before output to the plant (cf. figure 22). The sample rate of this system will be constrained to the sample rate of the sensor (camera), which is 25 Hz in its final implementation [10]. The discrete derivative is achieved with a finite difference calculation. Discrete simulations of both the linear and nonlinear feedback controllers are shown in figures 23 and 24. An important part of any PD controller design is dealing with signal noise, particularly in the derivative term. When a noisy signal is differentiated the noise is amplified. For this application, noise is expected to occur in the sensor output. It was found that using an *attenuating differentiator* $G(s) = \frac{\omega s}{s+\omega}$ in place of an ideal differentiator in the PD controller eliminated high-frequency noise when a cutoff frequency of $\omega = 5 rad/s$ was used. The equivalent z-domain transfer function for $G(s)$ is $G(z) = \frac{5z-5}{z-0.8187}$. The system was simulated using this filter in place of the discrete differentiator and figures 25 and 26 show the step responses of the two controllers that were designed.

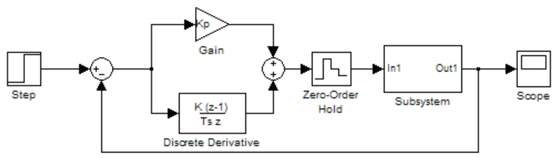


Figure 22. Discrete control strategy block diagram that is equivalent to the continuous-time design obtained earlier.

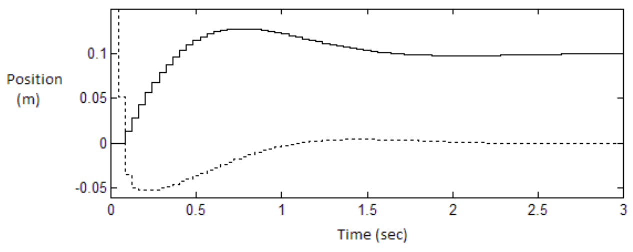


Figure 23. Simulated discrete step response of the linear PD control.

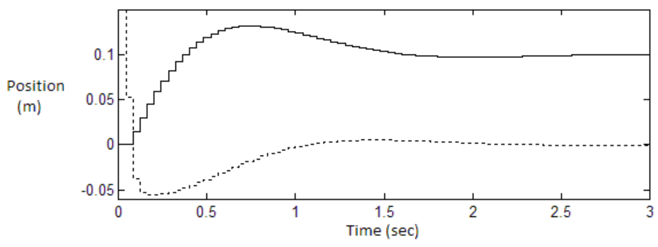


Figure 24. Simulated discrete step response of the feedback-linearized PD control.

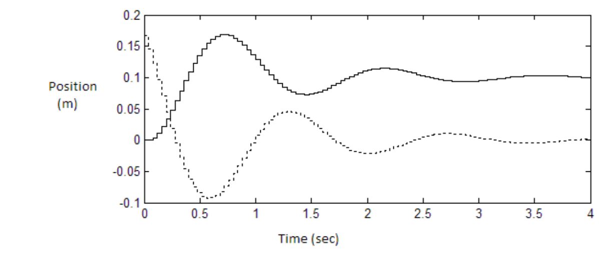


Figure 25. Simulated discrete step response of linear PD controller using filtered derivative.

9.1. Results

Both the linear and nonlinear feedback controllers were implemented on the test platform. The step response for linear PD feedback with $k_p = 0.52$ and $k_d = 0.23$ is shown in 27. The oscillations are pronounced, as predicted by the model due phase lag in the filtered differentiation. Tuning gains to $k_p = 1$ and $k_d = 1$ results in a more satisfactory response, shown in figure 28.
The step response for the feedback-linearized controller is shown in figure 29 for $k_p = 7.67$ and $k_d = 3.32$. Similarly, the oscillations are more pronounced than in the simulated model due to unmodeled delays and phase lag. Tuning the gains to $k_p = 10$ and $k_d = 10$ results in the response shown in figure 30.

9.2. Trajectory Tracking

As a final step, control gains were tuned for tracking a trajectory on the plate. Performance of the feedback-linearized controller is shown in figure 31 for tracking a slow circular trajectory with $k_p = 30$ and $k_d = 12$.
Increased control gains are required for slow trajectory tracking, since even a small error needs to be corrected. The difficulties here are the unmodeled nonlinearities in the system. These include:

1. The ball is not perfectly round. This causes unforseen changes in the required control effort.
2. Similarly, the plate is not perfectly flat. In fact, some miniscule convexity can be expected from the plate bending under its own weight.
3. The rolling ball has a dead zone due to static friction. For any angles less than approximately 1s, the ball does not move.

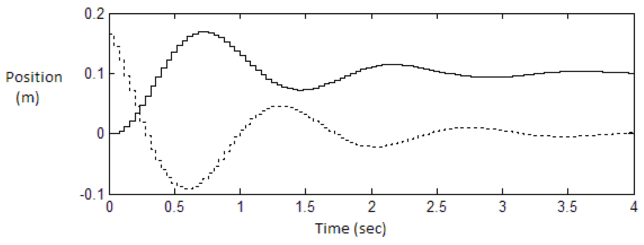


Figure 26. Simulated discrete step response of the feedback-linearized PD controller using the filtered derivative.

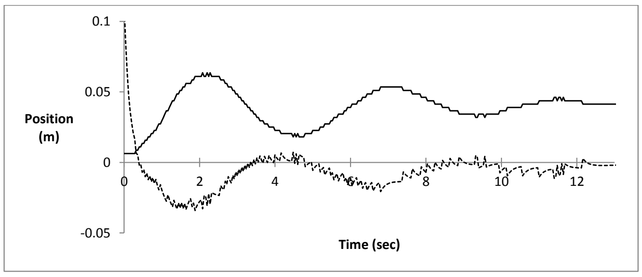


Figure 27. Experimental step response for linear PD control.

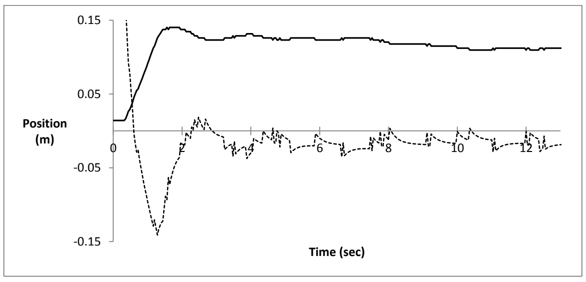


Figure 28. Experimental, tuned step response for linear PD control.

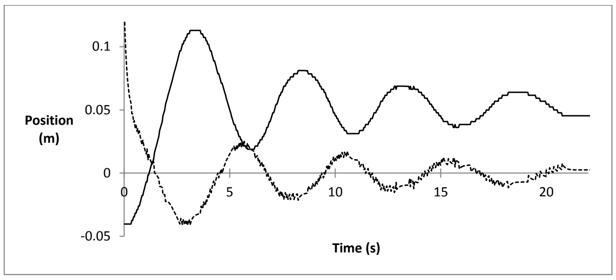


Figure 29. Experimental step response for feedback-linearized PD control.

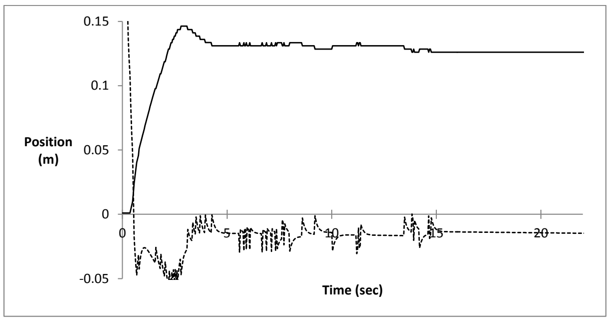


Figure 30. Experimental, tuned step response for feedback-linearized PD control.

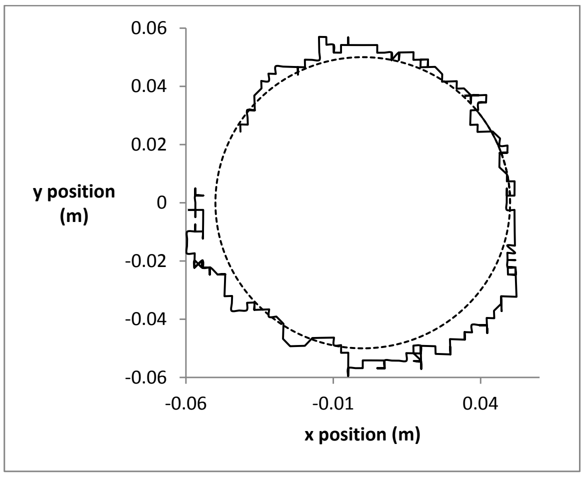


Figure 31. Circular trajectory tracking ball position.

4. Dynamic friction. The unmodeled rolling friction and air friction slow down the response of the ball.

5. Elastic deformation of the ball and plate at the point of contact with the plate.

10. Conclusions

A ball-on-plate balancing system was implemented using three Novint Falcon devices as actuators. PD control and feedback-linearized PD control both were found to be satisfactory. The model approximated the results well, demonstrating the viability of the Novint Falcon as an actuator for small-scale control applications. Some improvement in the performance of the system is possible by using a more spherical ball or flatter plate. The near-linear nature of the ball-on-plate system makes it an ideal candidate for linear optimal control design. It is possible to design a linear quadratic regulator to regulate the system to the origin. More specifically, a finite-horizon discrete-time linear quadratic regulator may be a good choice.

The nature of this ball-on-plate system required some constraints to be placed on the plate's position (see section 6.1). Some interesting problems could arise when these constraints are changed. For example, the ball's position on the plate can be controlled by a pure translation of the plate, rather than a change of angle. An exploration of these possibilities is left for future work.

The vision processing algorithms used are functional, but the ball position signal is subject to significant low-frequency noise, as well as some uncertainty. A better algorithm undoubtedly exists. Two possibilities are edge detection and optical flow. Edges are less sensitive to ambient light and color threshold changes. Optical flow produces a field of displacement vectors defining the translation of each pixel in a region. As a final thought, several algorithms could be used together with a Kalman filter to provide a converging estimate of the ball's position.

A short video of the system described in this paper can be found at this link (cf.. http://dl.dropboxusercontent.com/u/9864801/iWeb/Site˙2/Control˙Education˙&˙Applications˙2˙files/thesis.m4v).

References

[1] http://www.novint.com/index.php/products/novintfalcon.
[2] S. Martin and N. Hillier, "Characterization of the Novint Falcon Haptic Device for Application as a Robot Manipulator," in *Proc. Australasian Conference on Robotics and Automation* (ACRA), Sydney, Australia, December, 2009.
[3] L.W. Tsai, and R. Stamper, "A Parallel Manipulator with Only Translational Degrees of Freedom," Proceedings of the 1996 ASME Design Engineering Technical Conferences, Irvine, CA, 96-DETC/MECH-1152, 1996.
[4] L.W. Tsai, and R. Stamper, "A Parallel Manipulator with Only Translational Degrees of Freedom," ISR Technical Report TR-97-72, 1997.
[5] C.G. Broyden, "A class of methods for solving nonlinear simultaneous equations," Math Comput. 19 (1965), 577-593.
[6] M.T. Heath, Scientific Computing: An Introductory Survey, McGraw-Hill Series in Computer Science, McGraw-Hill, New York, 1997.
[7] Texas Instruments Inc., "TMS320F28335, TMS320F28334, TMS320F28332, TMS320F28235, TMS320F28234, TMS320F28232 Digital Signal Controllers (DSCs)," datasheet, 2007, Revised 2010.
[8] Allegro Microsystems Inc., "Full-Bridge PWM Motor Driver," A3953 datasheet, 2008.
[9] LSI/CSI, "32-Bit Quadrature Counter with Serial Interface," LS7366R datasheet, 2009
[10] OmniVision, "OV6620 Single-Chip CMOS CIF Color Digital Camera," OV6620 datasheet, 1999
[11] Texas Instruments Inc., "OMAP-L138 Low-Power Applications Processor," OMAP-L138 datasheet, 2009
[12] M.W. Spong, S. Hutchinson, and M. Vidyasagar, Robot Modeling and Control, Third edition, John Wiley, New York, ISBN 0-471-64990-2.
[13] R. R. Sumar, A. A. R. Coelho and L. D. Coelho, Computational intelligence approach to PID controller design using the universal model, Information Sciences, vol. 180, no. 20, pp. 3980-3991, 2010.
[14] S. Thomsen, N. Hoffmann and F. W. Fuchs, PI control, PI-based state space control and model based predictive control for drive systems with elastically coupled loads - A comparative study, IEEE

Transactions on Industrial Electronics, vol. 58, no. 8, pp. 3647-3657, 2011.

[15] M.-B. Radac, R.-E. Precup, E. M. Petriu and S. Preitl, Application of IFT and SPSA to servo system control, IEEE Transactions on Neural Networks, vol. 22, no. 12, pp. 2363-2375, 2011.

[16] W. Zhi, Q. S. Luo and J. F. Liu, An improved PID tuning algorithm for mobile robots, in: Advances in Electronic Commerce, Web Application and Communication, Volume 1, D. Jin and S. Lin (Eds.), Advances in Intelligent and Soft Computing, Springer-Verlag, vol. 148, pp. 345-353, 2012.

[17] E. Joelianto, D. C. Anura and M. P. Priyanto, ANFIS - hybrid reference control for improving transient response of controlled systems using PID controller, International Journal of Artificial Intelligence, vol 10, no. S13, pp. 88-111, 2013.