J1

Journal of Intelligent and Robotic Systems **29:** 277–294, 2000.

Robotic Optimization and Testing for the Formula One Tire-Changing Robot

RAUL MIHALI, MHER GRIGORIAN and TAREK SOBH
Department of Computer Science and Engineering, University of Bridgeport, Bridgeport, CT 06601, U.S.A.

(Received: 12 October 1999; in final form: 21 February 2000)

Abstract. Robotics experiences tremendous evolutions every year. Once a topic is mainly approached at research centers, read through highly specialized books and viewed distantly on scientific channels, nowadays it is a common and very approachable subject among undergraduate students of many universities. More and more robots are being designed every day, demanding technological implementation and production. This progress does not come without its glitches, however. A common and increasing problem that appears is the insufficient testing, simulation and optimization steps that a robotic construction needs to pass in order to achieve an efficient design. These steps prove to be difficult and sometimes discouraging, resulting in laborious work, due to lack of tools. This paper presents an example of a robotic optimization and testing, using a generic software package, applied on a custom manipulator, a tire-changing robot. Although the manipulator is designed with its own simulation and control package, it may lack optimality or validity. We implemented a different software package, focused on optimization and control of simple generic robots (XXX.RRR types) and apply the package on the tire-changer manipulator. The results provide improvements for the primary controlling software and confirm its correctness.

Key words: automation, manipulation, mechanical optimization, prototyping, robotics, sensing.

1. Introduction

The idea of a tire-changing robot arose after watching the dangerous and primitive tire changing process on Formula One type of racing cars. Normally a racing team sets up their own pit stand and is ready to service their car throughout the race. The racecars usually need one or more sets of tires during a race, and rarely need a more complex assistance (damages, defects, etc.). There are always well over 10 such pit stands, each having 15 to 25 qualified team members ready for very quick interventions.

A main problem associated with this setup is the serious accidents that sometimes occur. Having people change the tires of a car while almost in motion, after reaching critical pressure and temperature values is a risky challenge, an exposure to serious dangers. Because these stands are very close to the viewers, quite often they are also victims of such accidents (formula one literature provides plenty of examples – viewers killed by incorrectly mounted tires, team members run over by cars, explosions, etc.).

A second problem of these pit stops is the time differential between teams which substantially affects the racing quality. While some teams are capable of servicing the car in 6–7 seconds (usually replace four tires and load the car with fuel), others easily exceed 15 seconds and considering the speeds of these cars, such differences immediately affect the final classifications of the pilots.

Building an automated system to solve the issues above without introducing new ones involves exhaustive testing and optimization measures. Next, the tire-changing manipulator is being introduced: design aspects, various kinematics and dynamics issues. In the second part of the paper, the optimization and testing aspects of this design are discussed, the results being circulated through a second simulation package.

2. The Tire-Changing Robot

Our idea is to build a fully robotized system that takes over the tire-changing and refueling process. There is practically no need of human intervention. The system demonstrates remarkable time accuracy, precision and low risk implications. While our current design cannot achieve performances better than 10 to 15 seconds (in comparison with the 6–7 seconds actual records), it is capable of maintaining its exact time with a precision of milliseconds for all the cars, thus eliminating pit stop time variations.

2.1. BRIEF CONSIDERATIONS/ARMS/WORKSPACE

Our proposed robotic system consists of five manipulators: one for each of the tires, and a fifth one for the fuel tank. To preserve the environment of the pit stop and to assure the comfort of the team we implement suspended manipulators. The support of the five arms allows a sliding motion of each arm and does not create any obstacles or driving difficulties. The support has two double longitudinal branches on which the arms are suspended. The sliding mechanism of the arms is essential for the end effector positioning. The material used has to be resistant, of low elasticity, and capable of sustaining the mass of the arms. This paper does not discuss further about the refueling manipulator, however its implementation will be very similar.

Each of the tires of a Formula One car is fixed to the body with a single central screw (Figure 1). This design allows a flexible end-effector with decent power and mass requirements.

Each of the manipulators has a sliding range of 1 to 1.5 meters on the supports and can handle a tire in many ways. The only plane in which a good dexterity is required is the horizontal one (the distance from the ground and the tire's central axis is relatively constant). Based on the above-mentioned requirements, the manipulator design in Figure 2 has been derived.

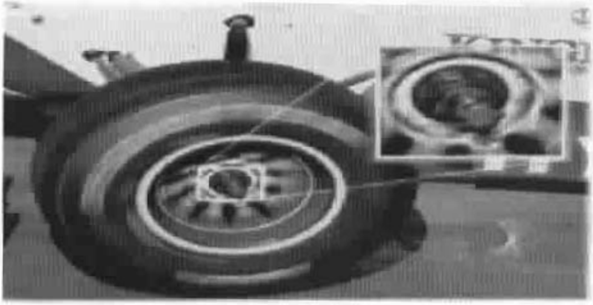


Figure 1. Tire close-up.

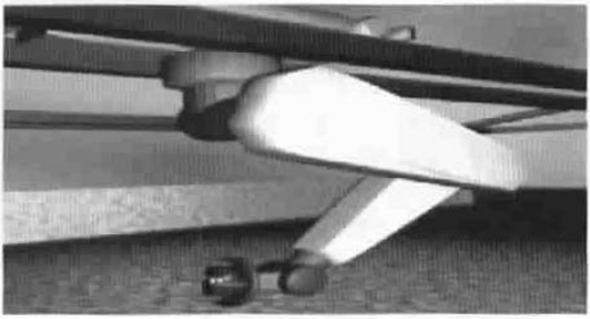


Figure 2. Arm view.

2.2. TASKS AND MOTION RELATED BRIEFINGS

The car arrives into the pit from a certain direction and stops in approximately the same spot every time. By the time the car arrives, its exact position and tire directions are registered. Once it stops and is jacked up, the arms start the *tire-changing process*. For lifting the car, a simple lifting system will be positioned on the stopping platform. Each manipulator has to go through the following task sequence:

- Position the end-effector as a function of the tire parameters received from the sensor system.
- Rotate the end-effector so that it can catch the tire.
- Grab the tire/remove the screw.
- Remove the tire from its axis and put it on the ground near the car in a convenient spot.
- Change position and grab a new tire, located in the proximity, with a new screw on it.
- Return and mount the new tire.
- Tighten the screw.
- Move back in the *stand-by* position/the car can go now.

There are about 15 different tasks, each of approximately 1 second, which allows a process length of approximately 10–15 seconds per manipulator (as men-

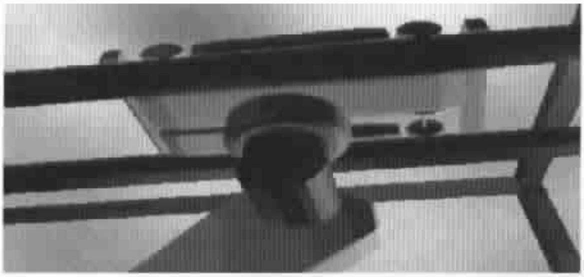


Figure 3. The slider.

tioned above, this design certainly does not reach current records, but conserves its performance and does not introduce time differences between racers). The arms work in parallel and independently. The set of movements required is of short distance and mainly consists of revolute steps: arm expansion/contraction, arm/end effector rotation and end effector positioning. There is a good chance that the specified time of around 1 second per move can be improved. According to the information from the tire-mounted sensors, the end effector can position itself on the axes of the tire and grab it correctly. The system can be easily adjusted to handle similar tires that are based on one screw. The rotation of the screw is a simple task, implying the activation of one compressed air tool with good dynamics control.

The most time consuming task is tire handling. This task requires good torque and acceleration control on the entire arm, the activation of all the engines, precision sliding [10, 20]. Moving back in the *stand-by* position is a simple task, to be completed partially when the car leaves. Because of the sliding mechanism, the pilot does not have to be precise while parking.

2.3. JOINT/LINK REQUIREMENTS AND CONSTRUCTION

One arm is composed of four joints and an end effector. The first joint is prismatic and constitutes the sliding part of the system (Figure 3).

Typically, the slider is activated at the beginning of the full process, to fix the arm in an appropriate position. The friction coefficient of sliding between the support and the slider has to be large enough to allow a stable braking with a precision of 1 m/s^2 and the friction coefficient of revolution has to be small enough for low acceleration control.

All the engines work at high speeds and have significant mass, and so the inertia problem has to be considered thoroughly [4, 9, 15]. To optimize the time, the arm moves from/to the stand-by position to/from the ready position at the same time with the sliding action. The second joint is revolute, as are all of the following ones. Figure 4 shows the joint and indicates the rotation direction.

A revolution limitation of 3/2 $*$ PI avoids kinetic or dynamic problems (e.g., singularities). The engine is fixed in the sliding part, thus concentrating the mass

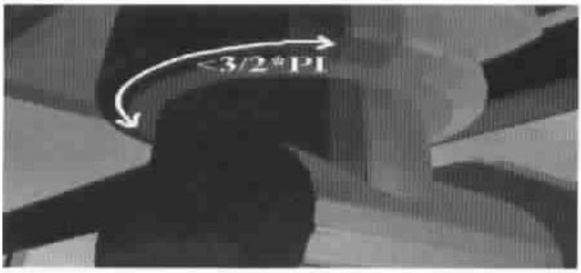


Figure 4. Second joint (bottom left view – car side).

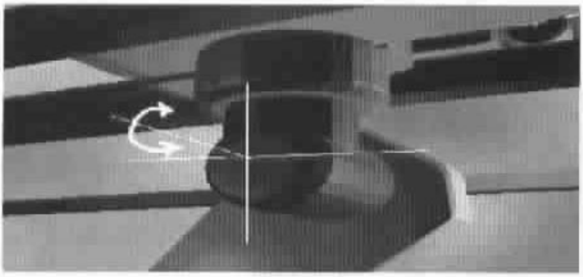


Figure 5. Third joint (view from car side).

pressure on the support. The third joint, together with the previous one and the slider, forms the *rigid* concentration of mass and torque of the arm (Figure 5).

For a better control, the engine has been attached to the axis of the previous joint. The rest of the arm is light and forms the *transportable* part, which needs to be fast. The angle of rotation has been limited to less than PI/2 degrees.

The last revolute joint from the arm segment is the elbow joint (Figure 6). This joint's engine has a moderate torque and is light. It is installed in the upper part of the arm, thus keeping a safe distribution of mass. The angle of revolution has been limited to PI/2 degrees. The pressure between the support and the arms has to be as small as possible, mostly because while the arms work together the support vibrations can force dislocations.

The fully extended position (at about PI for the third joint and PI/2 for the fourth joint) requires a special orientation of the end effector for not touching the ground. The stand-by position is safe enough to offer the pilot good visibility while entering the pits.

2.4. THE END EFFECTOR (DESIGN/POWER/ACCURACY)

The end effector has to be small, light, but powerful, dexterous and quick. After going through various models, we derived the design in Figure 7.

This model solves many problems. First, there are no position/orientation problems. The disk type effector can rotate at a speed ω_2 and reach any orientation requested by the sensor system. Having four identical grabbing segments, there will

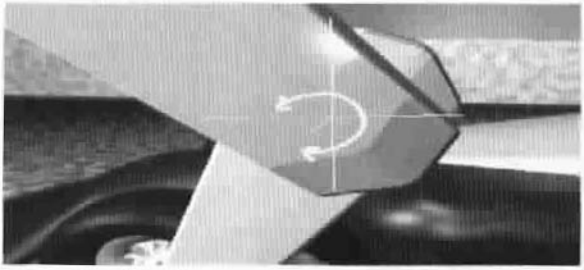


Figure 6. Elbow joint.

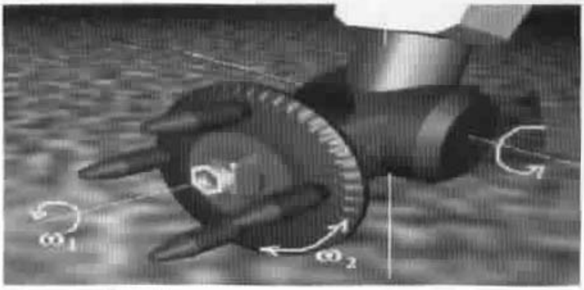


Figure 7. End effector.

be no equilibrium problems during transportation. The forces are well distributed and allow movements within a wide acceleration range. The revolute joint between the arm and the effector allows a rotation in the vertical plane of PI/2 degrees. The engine is light with moderate torque requirement. The engine that spins the disk with the four segments is installed in the pyramidal body following the cylinder, in the same spot as the compressed-air screw removal system.

The only rotation that cannot be performed by this end effector is on the vertical axis, however, this is compensated by the first revolute joint, which supports most of the torque requirements and allows for good acceleration control. In this setup, the end effector can operate for almost any reachable position of the tire. Another advantage of this effector model is that the tires do not have to be perpendicular to the ground (suppose an accident has happened, the end effector would still be able to accommodate the correct orientation). However, once the tire is not perpendicular to the ground this would mean that the car has been damaged seriously and most probably needs intervention of the team (the tire sensors prove very important here).

In order to find the position of the screw on the tire, the compressed air screwdriver starts a revolute task and at the same time advances slowly until it "fits" the faces of the screw and fixes onto it.

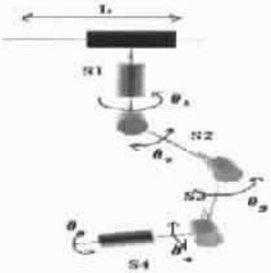


Figure 8. Manipulator scheme.

2.5. DIRECT AND INVERSE KINEMATICS

One of the next steps is solving the direct and inverse kinematics for this specific manipulator [19] (Figure 8).

Here, six joints of the arm can be seen. Using the Denavit–Hartemberg table [13], the equations for the direct kinematics can be written (the dimensions of the links are known):

$$\begin{aligned}
X &= L + \cos(\theta_1) * (S2 * \sin(\theta_2) - S3 * \sin(\theta_2 + \theta_3) - \\
&\quad - S4 * \sin(\theta_2 + \theta_3 + \theta_4 - \mathrm{PI}), \\
Y &= -\sin(\theta_1) * (S2 * \sin(\theta_2) - S3 * \sin(\theta_2 + \theta_3) - \\
&\quad - S4 * \sin(\theta_2 + \theta_3 + \theta_4 - \mathrm{PI}), \\
Z &= S1 + S2 * \cos(\theta_2) - S3 * \cos(\theta_2 + \theta_3) - S4 * \cos(\theta_2 + \theta_3 + \theta_4 - \mathrm{PI}), \\
\theta_x &= 0, \\
\theta_y &= 3 * \mathrm{PI}/2 - (\theta_2 + \theta_3 + \theta_4), \\
\theta_z &= \theta_1,
\end{aligned}$$

where x, y, z, are the coordinates and $\theta_x, \theta_y, \theta_z$ the orientations of the end effector.

Solving for the inverse kinematics using direct algebraic methods [17], we obtain the following model:

$$\begin{aligned}
L &= (Y/\tan(\theta_z)), \\
\theta_1 &= \theta_z, \\
\theta_3 &= \arccos((S2 * S2 + S3 * S3 - (X + S4 * \sin(PI/2 - \theta_y) * \cos(\theta_z) - \\
&\quad - (Y/\tan(\theta_z))) * (X + S4 * \sin(\mathrm{PI}/2 - \theta_y) * \cos(\theta_z) - \\
&\quad - (Y/\tan(\theta_z))) - (Z + S4 * \cos(\mathrm{PI}/2 - \theta_y) - S1) * \\
&\quad * (Z + S4 * \cos(\mathrm{PI}/2 - \theta_y) - S1))/(2 * S2 * S3)), \\
\theta_2 &= \arccos((-(S3 * \sin(\theta_3)) * ((X - (Y/\tan(\theta_z)))/\cos(\theta_1) + \\
&\quad + S4 * \sin(\mathrm{PI}/2 - \theta_y)) + (S2 - S3 * \cos(\theta_3)) * \\
&\quad * \mathrm{sqr}((S2 - S3 * \cos(\theta_3)) * (S2 - S3 * \cos(\theta_3)) + \\
&\quad + (S3 * \sin(\theta_3)) * (S3 * \sin(\theta_3)) - ((X - (Y/\tan(\theta_z)))/\cos(\theta_1) +
\end{aligned}$$

$$
\begin{aligned}
&+ S4 * \sin(\mathrm{PI}/2 - \theta_y)) * ((X - (Y/\tan(\theta_2)))/\cos(\theta_1) + \\
&+ S4 * \sin(\mathrm{PI}/2 - \theta_y))))/((S2 - S3 * \cos(\theta_3)) * \\
&* (S2 - S3 * \cos(\theta_3)) + (S3 * \sin(\theta_3)) * (S3 * \sin(\theta_3)))). \\
\theta_4 &= 3 * \mathrm{PI}/2 - \theta_y - \theta_2 - \theta_3.
\end{aligned}
$$

The metrics referred are shown in Figure 9:

C = the center of the tire, referred as Cx, Cy in the paper;
P = the XYZ coordinate of the end effector, currently on the axis of the tire;
λ = the distance from the center of the tire to the position P of the end effector;
l = the distance from the center of the tire to the projection of the center of the slider on the tire's central axis (CP);
d = the projection of CP onto the Ox axis, on which the slider is located;
H = the distance from the car to the Ox axis;
α = the angle between the tire and and the Oy axis.

For θ_1 (the first revolute join), the angle does not have to exceed PI degrees. In the initial position (stand-by), the angle will always be positioned at 0 degrees (use Figure 8 for the descriptions from this paragraph). The following three joints have been referred in terms of the previous link direction. For θ_2, the angle does not have to exceed PI/2, measured by the *s1* link. The angle would reach a value close to 0 degree (fully extended) very rarely, when the car is situated far from the arm, 80 cm or more. The initial position of this angle will be set close to PI/2, so the link will go up.

For θ_3, the reference to the previous link proves a superfluous allowance for the angle. We use values between PI/12 and up to PI, measured by the *s2* link. For the stand-by position, the angle will be set close to PI/12 (link goes up). θ_4 also has lower limits than physically possible. The angle value will not be smaller than PI/4 and no higher than $5 * \mathrm{PI}/4$, measured by the *s3* axis. Slightly larger angles (close to PI/4 or $5 * \mathrm{PI}/4$) would cause problems holding the tire, possibly touching the arm. A value of PI/2 is used for the stand-by position.

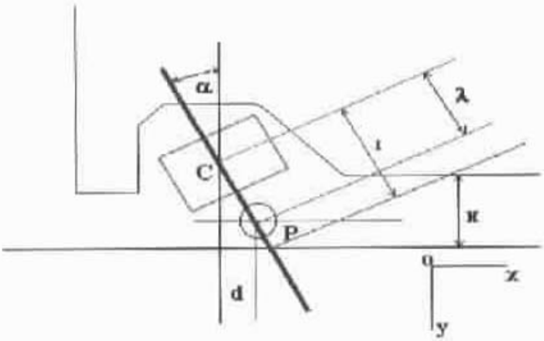


Figure 9. Front left tire scheme (top).

The last joint (θ_4) is adjusted independently of the others. The value can run from 0 up to 2 * PI. A software tracking system is being built, allowing rotation of the four segments synchronously from the moment the sensor system gives information about the tire's position. Thus, the angle can go up to n * PI. This might also allow positioning of the segments in advance.

Decoupling of singularities is not necessary as long as the design allows their avoidance [2, 11]. The inverse velocity and acceleration result from the following derivations:

$$\begin{aligned} \mathrm{d}q &= J(q)^{-1} * \mathrm{d}X, \\ \mathrm{d}^2 q &= J(q)^{-1} * b, \end{aligned}$$

where

$$\begin{aligned} B &= \mathrm{d}^2 X - \mathrm{d}/\mathrm{d}t * J(q) * \mathrm{d}q, \\ \mathrm{d}^2 X &= J(q) * \mathrm{d}^2 q + \mathrm{d}/\mathrm{d}t * J(q) * \mathrm{d}q, \end{aligned}$$

where:

q = the vector of joint coordinates;
$J(q)$, $J(q)^{-1}$ = the Jacobian and inverse Jacobian of q;
X = the vector of end effector coordinates.

2.6. DIRECT AND INVERSE DYNAMICS

For this type of arm the following dynamics model [7, 12, 15, 19] is used:

$$\begin{aligned} \tau &= M(q) * \mathrm{d}^2 q + V(q, \mathrm{d}q) + G(q) + F(q, \mathrm{d}q), \\ \mathrm{d}^2 q &= M^{-1}(q) * [\tau - V(q, \mathrm{d}q) - G(q) - F(q, \mathrm{d}q)], \end{aligned}$$

where:

τ = the end effector torque,
M = the symmetric joint-space inertia matrix,
V = describes *Coriolis* and *centripetal* effects [4, 12],
G = the gravity loading,
F = the end effector force.

2.7. THE SENSOR SYSTEM

The variable elements derived from the sensor system (that affect the inverse kinematics equations) were Cx, Cy (the center of the tire) and the angle α made by the axis of the tire with the slider. By receiving the XYZ coordinates of each of the four tires from the tire mounted positioning sensors, the above variable can be easily deducted.

There is also a need to control the number of times per second the sensor system provides data [17]. This is important to determine the car's motion. Motion recovery would allow one arm to track the tire and to have the end-effector positioned even before the car would stop, thus saving time [14].

According to the required sensor system tasks, one of the possible implementations for this sensory system can be through a radio radar detector. The receiving part of the system situated close to the scene will stay in stand-by mode and scan for signals from the tires. Once the receiver detects the sensors, this implies that the car is around, and according to the distance and the speed of the car, the software will process and send the necessary information to the arm controller.

Other tasks can be assigned to this system (i.e. analyzing the information from all the four tires, scanning the planarity of the car, vibrations, installation of new sensors providing different types of information, etc.). The sensor system's exact functioning, mounting and characteristics need a careful and detailed analysis, beyond the goal of this paper.

2.8. CONTROLLING AND SUPERVISING

The following parameters require continuous surveillance:

- Engine activation requests/request-reply discrepancy, internal functionality status.
- Link position/orientation, requested/resulted revolution angle difference, smoothness of revolution.
- Mass distribution in each arm, vibration factor evolution.
- Evolution of the delay in answering.
- Coordinate discrepancy between the sensor data and the actual position detected by the final effector.
- Sensor's displacement in time, sensor functionality.
- Support displacement, internal tension during arms motions, vibration and material response.
- Temperature and pressure of the environment and of the engines, wind velocity and direction.
- Parameter analysis evolution and general system status.

The required joints positions and orientations are always pre-simulated and compared with the ones obtained from the direct sensor output. The parameter difference is corrected using mostly PID control. For the digital feedback controllers, we consider a proportional plus derivative (PD) system [3, 6, 18], hoping to considerably simplify the nonlinear dynamic equations as well as to achieve a high update rate (Figure 10).

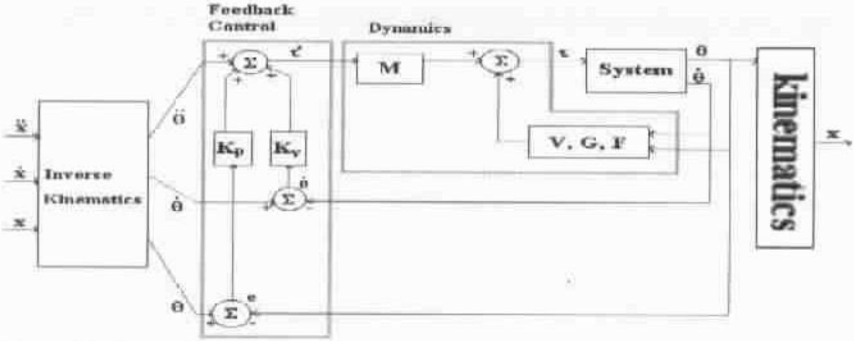


Figure 10. Dynamic simulation and control model.

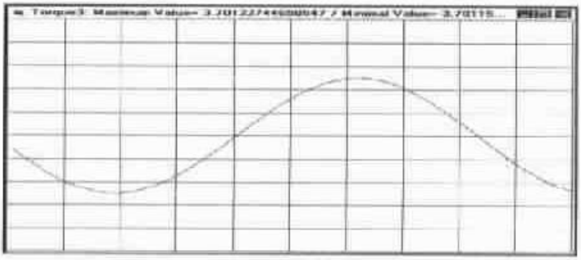


Figure 11. Torque 3 distribution in one second (N m).

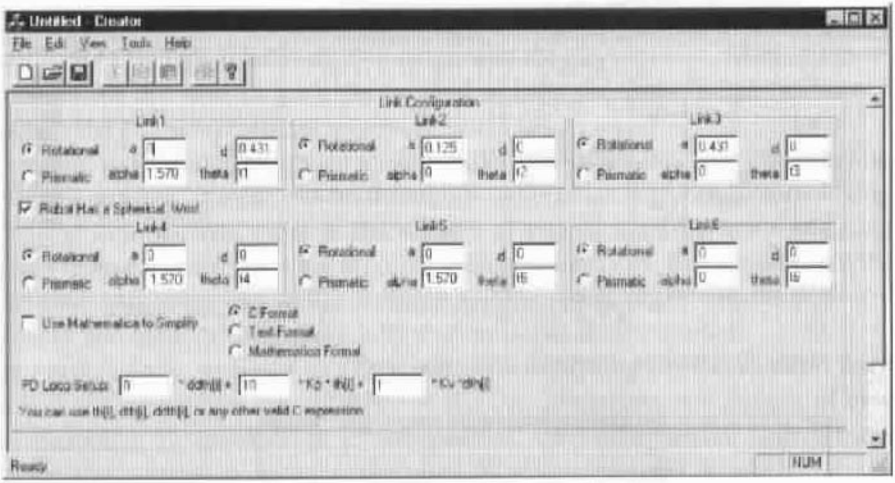
2.9. CURRENT DEVELOPMENT STAGE AND RESULTS

Currently the simulation/CAD module is connected with the kinematics and dynamics modules. Simulations showing the entire tire-changing process have been performed, too [16]. The next example (Figure 11) shows the torque applied in a joint for a *stand-by/full-extend* sequence, 1 second.

3. Control Analysis

It is mandatory to have an accurate simulation and control model before building the manipulator. In testing and optimizing our models, we have successfully developed and used a simulation package that accepts as input the configuration of a generic robot in D-H parameter form and the robot's dynamics parameters, and outputs a variety of closed form solutions that are essential to the design. The package also optimizes several control and structure parameters based on simulated task descriptions [3, 5]. Although our robot has already been designed with a simulation and control package (written in Visual Basic), it is critical to compare and eventually improve its results using a second or more packages like the one introduced here (written in Visual *C*++).

To turn our robot into a 6 DOF manipulator, we simply disregard the slider. This prismatic link has no trajectory interest and its position gets determined indepen-



Link Configuration

Link	Type	a	d	alpha	theta
Link1	● Rotational ○ Prismatic	1	0.431	1.570	t1
Link2	● Rotational ○ Prismatic	0.125	0	0	t2
Link3	● Rotational ○ Prismatic	0.431	0	0	t3
Link4	● Rotational ○ Prismatic	0	0	1.570	t4
Link5	● Rotational ○ Prismatic	0	0	1.570	t5
Link6	● Rotational ○ Prismatic	0	0	0	t6

☑ Robot Has a Spherical Wrist

□ Use Mathematica to Simplify

● C Format
○ Text Format
○ Mathematica Format

PD Loop Setup: 1 *ddth[i] + 10 *Kp*th[i] + 1 *Kv*dth[i]

You can use th[i], dth[i], ddth[i], or any other valid C expression

Figure 12. Input parameter window of the simulation-control package.

dently of the rest of the arm, as its center has to be in the vertical plane of the tire's axis (see 2.5. *Direct and Inverse Kinematics Approach*).

3.1. ROBOT SOLVING

The first part of the software package solves most of the robot motion module equations (Figure 12). For the direct kinematics, given the D-H parameter table, the package creates the $A_0^1, \dots, A_5^6$ matrices and obtains the $T_0^1, \dots, T_0^6$ transformation matrices in their symbolic form and base coordinates. For velocity kinematics, the software derives the Jacobian matrix and output equations in the form

$$\dot{X} = J * \dot{Q},$$

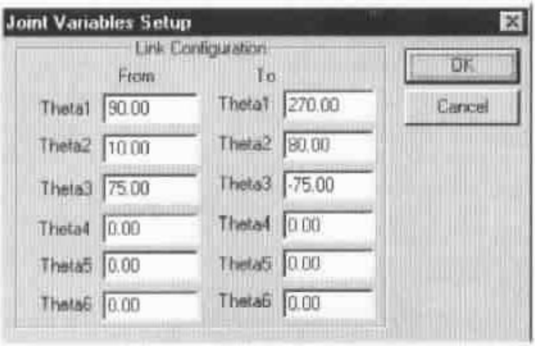
where:

$\dot{X}$ = the Cartesian velocity vector,
J = the Jacobian matrix,
$\dot{Q}$ = the joint velocity vector.

The package also implements a symbolic matrix inversion routine that solves the inverse velocity equations.

Trajectory plotting equations are also implementable in this software package, allowing for either cubic polynomial form or constant velocity with cubic polynomial blends. We have opted for the second choice. Three time intervals can be chosen: a $t_0 - t_1$ interval for an accelerating motion, a $t_2 - t_3$ interval for decelerating motion and a $t_1 - t_2$ for a linear/constant velocity period.

Given the M, G, and V matrices (see 2.4. *Direct and Inverse Dynamics Approach*), the package outputs the dynamics and inverse dynamics equations, too. All of the equations are generated in text, C/C++ source code, or Mathematica



Link Configuration

From		To	
Theta1	90.00	Theta1	270.00
Theta2	10.00	Theta2	80.00
Theta3	75.00	Theta3	-75.00
Theta4	0.00	Theta4	0.00
Theta5	0.00	Theta5	0.00
Theta6	0.00	Theta6	0.00

Figure 13. Setting up the thetas for the PD loop.

format, thus they are easy to be used for further scaling. Once we have obtained the necessary output from the package, testing in parallel with our software revealed a clear consistency.

3.2. SIMULATION AND OPTIMIZATION

The second part of the software package, also called the execution module, was of particular use for our tire changer. The package implements a local PD controller on the control function:

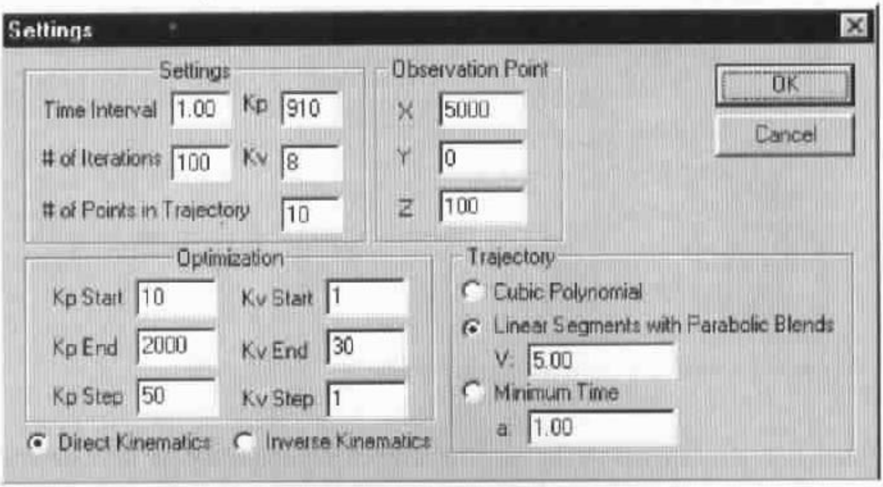
$$\tau = f(Q, \dot{Q}, \ddot{Q}) * \ddot{Q} + f(Q, \dot{Q}, \ddot{Q}) * K_p * e_p + f(Q, \dot{Q}, \ddot{Q}) * K_v * e_v,$$

where:

f = a function of the robot joint position, velocity and acceleration vectors (note that the package allows for any mathematical function),
$\ddot{Q}$ = the desired link acceleration vector,
K_p = the proportional gain,
K_v = the derivative gain,
e_p = the error in the joint variable vector,
e_v = the error in the joint velocities vector.

The package also allows us to add a PID or other feedback control functions [3, 14]. After providing K_p, K_v, the initial and final positions, the time interval in which the movement should be committed, the number of iterations in the PD loop and the trajectory generator to be used, the package will run the control loop at points specified by the user and output graphs showing the ideal and the real trajectories of the manipulator, plus searching and optimizing the K_p, K_v and update values.

As an example, we run the control loop on a move similar to the fully compressed – fully extended one for which we have included the torque graphs (Figures 13, 14), one second time limit.



Settings

Settings

Time Interval 1.00 Kp 910

# of Iterations 100 Kv 8

# of Points in Trajectory 10

Observation Point

X 5000

Y 0

Z 100

Optimization

Kp Start 10 Kv Start 1

Kp End 2000 Kv End 30

Kp Step 50 Kv Step 1

● Direct Kinematics ○ Inverse Kinematics

Trajectory

○ Cubic Polynomial

● Linear Segments with Parabolic Blends

V: 5.00

○ Minimum Time

a: 1.00

Figure 14. Additional PD data for the loop control.

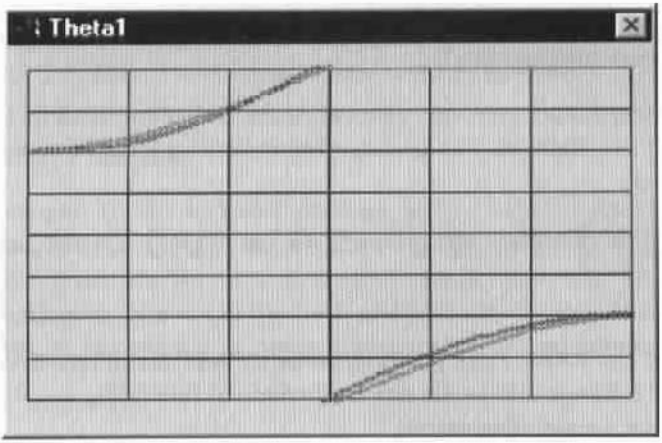


Figure 15. θ_1/time (desired versus actual).

As output, we can see the desired thetas versus the actual ones. The package also provided the optimal K_p and K_v values, after looping in the interval and parameter ranges (Figures 15, 16 and 17).

Making use of the trajectory plotting part of the program, we were able to script the output into our model and analyze the trajectories on the move. It is of major importance to have all the trajectories of the manipulator well defined. Figure 18 shows the trajectories of the move described above: the trajectory of the upper arm, elbow, and wrist.

The following are examples of other trajectories that have been successfully improved by means of the package (Figures 19, 20, 21, showing trajectories built with the simulator for the tire changer versus the improved ones built with the general simulation package). Note that most of the optimizations are due to improved

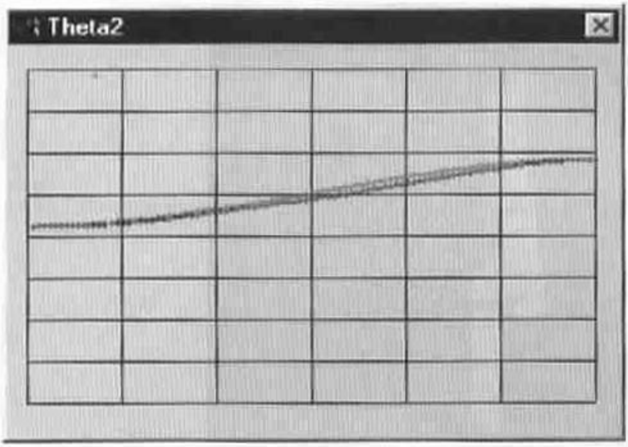


Figure 16. θ_2/time (desired versus actual).

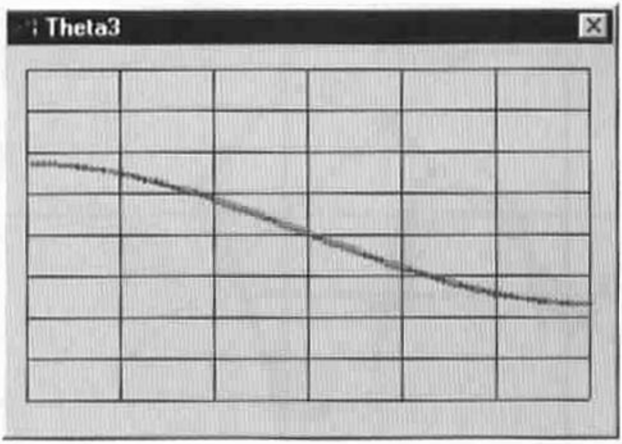


Figure 17. θ_3/time (desired versus actual).

values for K_p and K_v obtained through the generic simulation package. Also, note the relatively small differences between the output of the two packages, which confirms the validity of the tire changer simulation package.

Most of the simulation sequences discussed here are available as AVI movies on the website http://www.bridgeport.edu/~risc. Using the second software package, we were able to analyze and improve the whole movement that the tire-changer performs while changing a tire.

4. Conclusions

The design and symbolic construction of a robot can promote critical problems if not analyzed thoroughly before presented to manufacturing hands. It is much more expensive to discover a design flaw after the arm is actually built. Many

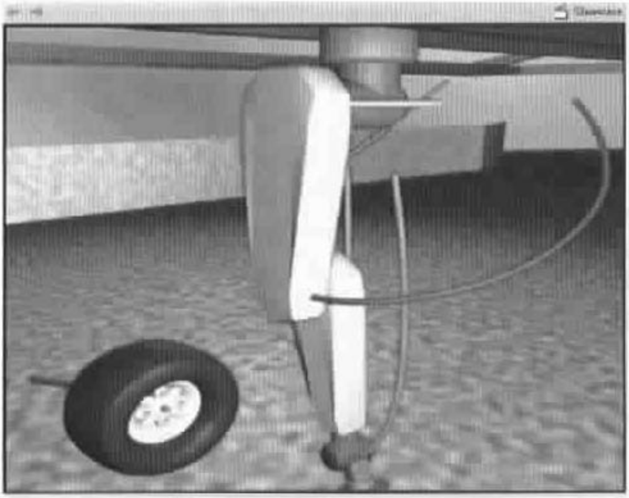


Figure 18. Plotted trajectories.

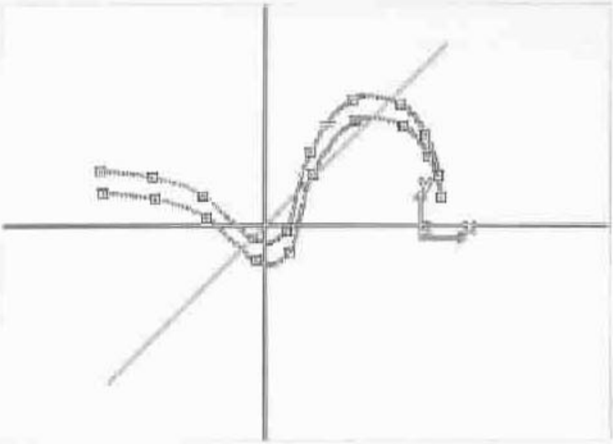


Figure 19. End Effector trajectory analysis 1.

times inconsistencies at that level proved to be unrecoverable, hence the importance of having as many optimizations and improvements as possible at the design and simulation phase.

In this paper, we design a complex manipulator and show the importance and advantages of sufficient optimization and testing. Having applied a generic simulation and optimization software over the main controlling/simulating package, we were able not only to confirm, but also to improve the results consistently. Trajectories have been adjusted (allowing for a lower inertial load on the arm and for faster reach of its destinations), the amount of necessary torque has been reduced and, most important the correctness of the manipulator's controlling software has been tested and corroborated.

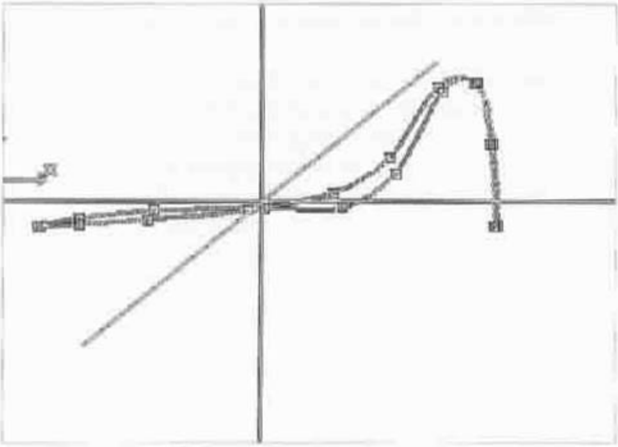


Figure 20. End Effector trajectory analysis 2.

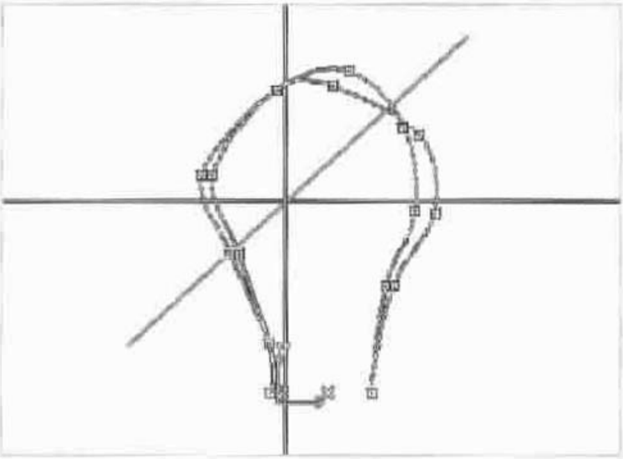


Figure 21. End Effector trajectory analysis 3.

References

1. Allen, P. K.: *Robotic Object Recognition Using Vision and Touch*, Kluwer Academic Publishers, Norwell, MA, 1987.
2. Banks, M. J. and Cohen, E.: Realtime B-spline curves from interactively sketched data, in: *Proceedings of the 1990 Symposium on Interactive 3-D graphics*, ACM, March 1990.
3. Benedetti, R. and Risler, J. J.: In: *Real Algebraic and Semi-Algebraic Sets*, Hermann, 1990, pp. 8–19.
4. Christie, D. E.: *Intermediate College Mechanics*, McGraw-Hill, 1952.
5. Craig, J. J.: *Introduction to Robotics. Mechanics and Control*, 2nd edn, Addison-Wesley, 1989.
6. Dekhil, M., Sobh, T. M., Henderson, T. C., and Mecklenburg, R.: UPE: Utah prototyping environment for robot manipulators, in: *Proceedings of the IEEE International Conference on Robotics and Automation*, Nagoya, Japan, May 1995.
7. Dekhil, M., Sobh, T. M., Henderson, T. C., Sabbavarapu, A., and Mecklenburg, R.: *Robot Manipulator Prototyping (Complete Design Review)*, University of Utah, 1994.

8. De Wit, C. C., Siciliano, B., and Bastin, G.: *Theory of Robot Control*, Springer-Verlag, London, 1996.
9. Herrea-Beneru, L., Mu, E., and Cain, J. T.: *Symbolic Computation of Robot Manipulator Kinematics*, Department of Electrical Engineering, University of Pittsburgh.
10. Hervé, J., Cucka, P., and Sharma, R.: Qualitative visual control of a robot manipulator, in: *Proceedings of the DARPA Image Understanding Workshop*, September 1990.
11. Li, Y. and Wonham, W. M.: Controllability and observability in the state-feedback control of discrete-event systems, in: *Proc. Conf. on Decision and Control*, 1988.
12. Marris, A. W. and Stoneking, C. E.: *Advanced Dynamics*, McGraw-Hill, 1967.
13. McKerrow, Ph. J.: *Introduction to Robotics*, Addison-Wesley, 1991.
14. Mihali, R. C. and Sobh, T. M.: The formula one tire changing robot (F1-T.C.R.), accepted for publication in the *J. Intell. Robotic Systems*, April 1999.
15. Nakamura, Y.: *Advanced Robotics – Redundancy and Optimization*, Addison-Wesley, 1991.
16. Rieseler, H. and Wahl, F. M.: Fast symbolic computation of the inverse kinematics of robots, Institute for Robotics and Computer Control, Technical University of Braunschweig.
17. Schalkoff, R. J.: *Digital Image Processing and Computer Vision*, John Wiley, Inc., 1989.
18. Sobh, T., Dekhil, M., Henderson, T. C., and Sabbavarapu, A.: Prototyping a three-link robot manipulator. Presented in the Second World Automation Congress, *Sixth International Symposium on Robotics and Manufacturing (ISRAM 96)*, Montpellier, France, May 1996.
19. Spong, W. M.: *Robot Dynamics and Control*, John Wiley, 1989.
20. Thingvold, J. A. and Cohen, E.: Physical modeling with B-spline surfaces for interactive design and animation, in: *Proceedings of the 1990 Symposium on Interactive 3-D Graphics*, ACM, March 1990.