AN INDUSTRIAL ROBOTIC SYSTEM
FOR MOVING OBJECT INTERCEPTION USING
IDEAL PROPORTIONAL NAVIGATION GUIDANCE

by

Jonathan M. Borg

A thesis submitted in conformity with the requirements
for the Degree of Master of Applied Science
Graduate Department of Mechanical and Industrial Engineering
University of Toronto

© Copyright by Jonathan M. Borg, 2000
The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author’s permission.

L’auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L’auteur conserve la propriété du droit d’auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-54104-5
To Anabel
for her love and patience.

&

To Raquelle
whose early days of life I’ve missed to complete this thesis.
Acknowledgements

I would like to express my sincere gratitude to Prof. B. Benhabib, my supervisor, for his continuous guidance, encouragement and support all throughout the thesis, and also for following me through my course of studies at the University of Toronto with great interest.

I am grateful to Mehran Mehrandezh, who developed the navigation-based interception technique that forms the basis of this thesis. Without his work, this thesis would not have been possible. I also express my gratitude to Michael Naish, for his valuable comments during the weekly laboratory meetings, to Yaron Derman, for his work on camera calibration, and to Christine Munroe, for her help in editing the final draft of the thesis.

I also wish to thank all my colleagues and friends in the Computer Integrated Manufacturing Laboratory at the University of Toronto, in particular Alejandro Ramirez, Martin Bonert, Mark Haberer, Shi-Chu Zhu and Maurizio Ficocelli. Their friendship and support were most important to help me in adjusting myself to student-life at the University.

My studies at the University of Toronto would not have been possible without the financial support of the International Council for Canadian Studies, and, for this, I am most grateful.

A final thanks is certainly due to my family for their continuous encouragement and support, alas from far away in Malta. Last, but certainly not least, I would like to express a special word of thanks to my wife, Anabel, for her unwavering support, patience and encouragement.
Abstract

An Industrial Robotic System for Moving Object Interception
Using Ideal Proportional Navigation Guidance

Jonathan M. Borg
M.A.Sc.

Department of Mechanical and Industrial Engineering
University of Toronto
2000

The problem area under study in this thesis is the development and implementation of a robot-motion planning technique to bring the end-effector of an industrial manipulator to a (pre-grasp) rendezvous location with a randomly-moving target in minimum time.

The robot-motion planning technique utilized herein is a hybrid interception scheme. The end-effector is initially controlled via a navigation-based method and subsequently, in the second phase, via a conventional tracking method that takes over control to ensure a smooth grasp. The proposed method is based on the navigation-based interception scheme originally developed by Mehrandezh et al. in [3]. In this thesis, however, these motion-planning
algorithms were modified to suit the industrial manipulator available in the Computer Integrated Manufacturing Laboratory.

The computer-simulation and experimental results demonstrated the capability of the proposed IPNG-based interception scheme to intercept slow- and fast-maneuvering targets faster than a pure tracking method would do.
  3.1 Introduction ................................................................................................. 28
  3.2 The GMFanuc S-100 Manipulator ................................................................. 28
  3.3 Kalman Filter Prediction ........................................................................... 30
    3.3.1 The Kalman Filter (KF) ........................................................................... 31
    3.3.2 Object-Motion Estimation via Kalman Filter ........................................ 32
  3.4 Robot-Motion Planning ............................................................................. 33
    3.4.1 Robot-Motion Initialization ................................................................. 33
    3.4.2 End-Effector Acceleration Command ................................................. 34
    3.4.3 Target Tracking .................................................................................. 37
    3.4.4 Switching from Navigation-Guidance to Tracking ............................. 40
    3.4.5 Overall Robot-Motion Planning Algorithm ....................................... 42
  3.5 Interception-Planning Simulations ............................................................. 43
  3.6 Summary ................................................................................................... 48

  4.1 Introduction ................................................................................................ 49
  4.2 System Configuration ............................................................................... 49
    4.2.1 Object-Tracking Sub-System ............................................................... 50
    4.2.2 Interception-Planning Sub-System .................................................... 52
    4.2.3 Execution Sub-System ....................................................................... 52
    4.2.4 Object Motion ................................................................................... 53
    4.2.5 System Integration ............................................................................ 53
      4.2.5.1 Communication Modules ........................................................... 55
4.2.5.2 System Initiation ................................................................. 55
4.2.5.3 System Delays ................................................................. 55
4.3 Experimental Results ............................................................. 56
  4.3.1 Slow-Maneuvering objects ................................................. 56
  4.3.2 Fast-Maneuvering objects ............................................... 61
4.4 Summary ............................................................................. 64

5. Discussion ............................................................................. 65
  5.1 Introduction ........................................................................ 65
  5.2 Effect of Unavailability of Robot’s Dynamics on Robotic IPNG ............................................. 65
  5.3 Navigation-Based Tracking - Motivation .................................................. 66
  5.4 Modified Quintic-Polynomial Tracking Method ........................................... 67
    5.4.1 Basic Concept ................................................................ 67
    5.4.2 Formulation of the Tracking Problem ........................................... 68
      5.4.2.1 Mapping position, Velocity and Acceleration Vectors ...................... 70
      5.4.2.2 Generating the Trajectories Along the $e_r$ and $e_\theta$ Axes ............. 71
    5.4.3 Switching from Navigation to Tracking ........................................... 73
    5.4.4 Simulation Example ................................................................ 74
    5.4.5 Experimental Results .......................................................... 76
      5.4.5.1 Slow-Maneuvering Targets ................................................... 77
      5.4.5.2 Fast-Maneuvering Targets .................................................. 79
    5.4.6 Discussion ......................................................................... 81
  5.5 Summary ............................................................................. 82

6. Conclusion ............................................................................. 83
  6.1 Summary & Conclusions ...................................................... 83
  6.2 Recommendations ................................................................ 85

References ................................................................................ 87
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1a</td>
<td>Planning of time-optimal robot trajectories in APPE</td>
<td>4</td>
</tr>
<tr>
<td>1.1b</td>
<td>Selection of optimal interception point using a travel / arrival time graph in APPE</td>
<td>4</td>
</tr>
<tr>
<td>1.2</td>
<td>IPNG-based moving-object-interception system</td>
<td>11</td>
</tr>
<tr>
<td>2.1</td>
<td>Interception geometry</td>
<td>14</td>
</tr>
<tr>
<td>2.2</td>
<td>Interception of a moving target via IPNG</td>
<td>17</td>
</tr>
<tr>
<td>2.3</td>
<td>Modifying the computed acceleration command</td>
<td>21</td>
</tr>
<tr>
<td>2.4</td>
<td>Schematic diagram of the simulated two-link manipulator</td>
<td>23</td>
</tr>
<tr>
<td>2.5</td>
<td>Interception of a target moving on a sinusoidal path</td>
<td>24</td>
</tr>
<tr>
<td>2.6</td>
<td>Position and velocity variation versus time for end-effector and target</td>
<td>24</td>
</tr>
<tr>
<td>2.7</td>
<td>Variation of the interception time versus switching time for the example depicted in Figure 2.5</td>
<td>25</td>
</tr>
<tr>
<td>2.8</td>
<td>Interception of a target moving on a parabolic path</td>
<td>26</td>
</tr>
<tr>
<td>2.9</td>
<td>Position and velocity variation versus time for end-effector and target</td>
<td>26</td>
</tr>
<tr>
<td>3.1</td>
<td>Architectural organization of GMF S-100 control</td>
<td>29</td>
</tr>
<tr>
<td>3.2</td>
<td>Upgrading $a_{IPNG}$ by adding a component in LOS direction (planar case)</td>
<td>35</td>
</tr>
<tr>
<td>3.3</td>
<td>Modifying the $a_{IPNG}$ due to violation of the velocity limit</td>
<td>36</td>
</tr>
<tr>
<td>3.4</td>
<td>Modifying the $a_{IPNG}$</td>
<td>37</td>
</tr>
<tr>
<td>3.5</td>
<td>Selection of the switching instant</td>
<td>41</td>
</tr>
<tr>
<td>3.6</td>
<td>Interception of a target moving on a sinusoidal trajectory (planar view)</td>
<td>43</td>
</tr>
<tr>
<td>3.7</td>
<td>Position and velocity variation versus time for end-effector and target</td>
<td>44</td>
</tr>
<tr>
<td>3.8</td>
<td>Variation of $t_{int}$ and $t_{qp}$ versus switching time for the example depicted in Figure 3.6</td>
<td>45</td>
</tr>
<tr>
<td>3.9</td>
<td>Interception of a target moving on a linear trajectory</td>
<td>46</td>
</tr>
<tr>
<td>3.10</td>
<td>Position and velocity variation versus time for end-effector and target</td>
<td>46</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>Figure 3.11</td>
<td>Variation of $t_{int}$ and $t_{aq}$ versus switching time for the example depicted in Figure 3.9</td>
<td></td>
</tr>
<tr>
<td>Figure 4.1</td>
<td>Architecture of the experimental IPNG-QPT interception system</td>
<td></td>
</tr>
<tr>
<td>Figure 4.2</td>
<td>Flowchart for the hybrid IPNG-QPT interception system</td>
<td></td>
</tr>
<tr>
<td>Figure 4.3</td>
<td>Interception of a target moving on a linear trajectory (planar view)</td>
<td></td>
</tr>
<tr>
<td>Figure 4.4</td>
<td>Position and velocity variation versus time for the end-effector and the target</td>
<td></td>
</tr>
<tr>
<td>Figure 4.5</td>
<td>Variation of $t_{int}$ and $t_{aq}$ versus switching time for the example depicted in Figure 4.3</td>
<td></td>
</tr>
<tr>
<td>Figure 4.6</td>
<td>Interception of a target moving on a circular trajectory (planar view)</td>
<td></td>
</tr>
<tr>
<td>Figure 4.7</td>
<td>Position and velocity variation versus time for the end-effector and the target</td>
<td></td>
</tr>
<tr>
<td>Figure 4.8</td>
<td>Variation of $t_{int}$ and $t_{aq}$ versus switching time for the example depicted in Figure 4.6</td>
<td></td>
</tr>
<tr>
<td>Figure 4.9</td>
<td>Interception of a target moving on a circular trajectory and changing its direction of motion (planar view)</td>
<td></td>
</tr>
<tr>
<td>Figure 4.10</td>
<td>Position and velocity variation versus time for the end-effector and the target</td>
<td></td>
</tr>
<tr>
<td>Figure 4.11</td>
<td>Variation of $t_{int}$ and $t_{aq}$ versus switching time for the example depicted in Figure 4.9</td>
<td></td>
</tr>
<tr>
<td>Figure 4.12</td>
<td>Interception of a target travelling along a stop-and-go trajectory (planar view)</td>
<td></td>
</tr>
<tr>
<td>Figure 4.13</td>
<td>Position variation versus time for the end-effector and the target</td>
<td></td>
</tr>
<tr>
<td>Figure 5.1</td>
<td>Relative velocity parallel to the LOS</td>
<td></td>
</tr>
<tr>
<td>Figure 5.2</td>
<td>Geometry of the modified Quintic-Polynomial tracking method</td>
<td></td>
</tr>
<tr>
<td>Figure 5.3</td>
<td>Mapping between the $xy$ - plane and the $e_xe_y$ - plane</td>
<td></td>
</tr>
<tr>
<td>Figure 5.4</td>
<td>Variation of $t_{int}$ versus switching time for a sinusoidal target trajectory</td>
<td></td>
</tr>
<tr>
<td>Figure 5.5a</td>
<td>End-effector's and target’s trajectories: switching at 5.7</td>
<td></td>
</tr>
<tr>
<td>Figure 5.5b</td>
<td>End-effector's and target’s trajectories: switching at 2.6 s</td>
<td></td>
</tr>
</tbody>
</table>
Figure 5.6  Position and velocity variation versus time for the end-effector and
the target ................................................................................................................. 76
Figure 5.7  End-effector and target trajectories (linear target trajectory) ............. 77
Figure 5.8  Position variation versus time for the end-effector and the target .......... 78
Figure 5.9  Variation of the interception time versus the switching time ............... 78
Figure 5.10 End-effector and target trajectories (circular target trajectory) .......... 79
Figure 5.11 Variation of the interception time versus the switching time ............... 79
Figure 5.12 Interception of a target moving on a circular trajectory and changing
its direction of motion ................................................................................................. 80
Figure 5.13 Position variation versus time for the end-effector and the target .......... 80
Figure 5.14a Interception of a target which changes its trajectory using the
hybrid IPNG-mod-QPT method ................................................................................. 81
Figure 5.14b Interception of a target which changes its trajectory using the
hybrid IPNG-QPT method ......................................................................................... 81
Nomenclature

LATIN LETTERS

\( a_c \) Acceleration command.
\( a_{eqv} \) Acceleration command which produces an end-effector velocity in the same direction as that produced by \( a_{IPNG} \).
\( a_{IPNG} \) Acceleration command of the IPNG.
\( A_{lim} \) Maximum robot end-effector acceleration.
\( A_{lim}, A_{lim r} \) Maximum robot end-effector acceleration divided along the \( e_r \) and \( e_\theta \) directions, respectively.
\( a_{PNG} \) Acceleration command of the PNG.
\( a_x, a_y \) Target’s \( x \) and \( y \) Cartesian acceleration, respectively.
\( C(q, \dot{q}) \) Coriolis acceleration in robot’s dynamic model.
\( c_0, c_1, c_2, c_3, c_4, c_5 \) Coefficients describing a polynomial function.
\( d \) Displacement vector.
\( e_\theta \) Unit vector forming a polar coordinate system with \( e_r \).
\( e_\phi \) Unit vector perpendicular to LOS and to \( e_z \).
\( e_d \) Unit vector defining the direction of robot motion-initialization.
\( e_r \) Unit vector in LOS direction.
\( e_{xt} \) Unit vector along the target’s velocity.
\( e_z \) Unit vector in a direction opposite to \( z \).
\( G(q) \) Gravitational force vector.
\( H_k \) Kalman Gain matrix.
\( I \) Identity matrix.
\( J(q) \) Jacobian of the robot.
\( K \) Factor used to limit the acceleration command of the IPNG.
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k$</td>
<td>Coefficient used to scale the LOS component when computing the direction of robot-motion initialization.</td>
</tr>
<tr>
<td>$K_p$, $K_d$</td>
<td>Proportional and derivative gains.</td>
</tr>
<tr>
<td>$L$</td>
<td>Direction normal to the acceleration command in the PNG-based navigation laws.</td>
</tr>
<tr>
<td>$M$</td>
<td>Measurement matrix relating the observed measurement to the system state.</td>
</tr>
<tr>
<td>$M(q)$</td>
<td>Inertia matrix of the robotic manipulator.</td>
</tr>
<tr>
<td>$P(q)$</td>
<td>Vectorial expression for the forward kinematics of the robotic manipulator.</td>
</tr>
<tr>
<td>$p_x$, $p_y$</td>
<td>Target’s $x$ and $y$ Cartesian position, respectively.</td>
</tr>
<tr>
<td>$q$</td>
<td>Robot’s joint state.</td>
</tr>
<tr>
<td>$q_x(t)$, $q_y(t)$, $q_z(t)$</td>
<td>Polynomial trajectories describing the motion of the end-effector in the $x$, $y$ and $z$ directions, respectively.</td>
</tr>
<tr>
<td>$R$</td>
<td>Operator describing the rotational transformation between two coordinate frames.</td>
</tr>
<tr>
<td>$r$</td>
<td>Relative positional vector between the target and the end-effector.</td>
</tr>
<tr>
<td>$\hat{r}$</td>
<td>Unit relative positional vector between the end-effector and the target.</td>
</tr>
<tr>
<td>$s$</td>
<td>Fading memory factor used by Kalman filter to discard old data.</td>
</tr>
<tr>
<td>$S_k$</td>
<td>System covariance matrix.</td>
</tr>
<tr>
<td>$t$</td>
<td>The system time.</td>
</tr>
<tr>
<td>$T$</td>
<td>The vector of joint actuator torques.</td>
</tr>
<tr>
<td>$\tilde{t}_{CT}$</td>
<td>Estimation of the time during which the robot is under the control of the CT-method.</td>
</tr>
<tr>
<td>$t_{int}$</td>
<td>Interception time.</td>
</tr>
<tr>
<td>$\tilde{t}_{int}$</td>
<td>Estimation of the interception time.</td>
</tr>
<tr>
<td>$t_{IPNG}$</td>
<td>Time during which the robot is under the control of the IPNG.</td>
</tr>
<tr>
<td>Symbol</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>$t_{modQP}$, $t_{modQP_r}$</td>
<td>Trajectory motion time of a one-dimensional trajectory describing the motion of the end-effector in the $e_\phi$ and $e_r$ directions, respectively.</td>
</tr>
<tr>
<td>$tol_p$, $tol_v$</td>
<td>Interception tolerances in position and velocity.</td>
</tr>
<tr>
<td>$t_{pq}$</td>
<td>Quintic-polynomial trajectory motion time.</td>
</tr>
<tr>
<td>$\tilde{t}_{qp}$</td>
<td>Smoothed value of the quintic-polynomial trajectory motion time computed on-line.</td>
</tr>
<tr>
<td>$t_{sw}$</td>
<td>The instant at which control of the end-effector changes from the navigation-based technique to the tracking technique.</td>
</tr>
<tr>
<td>$t_{Tracking}$</td>
<td>Time during which the robot is under the control of a tracking technique.</td>
</tr>
<tr>
<td>$t_{vel}$, $t_{acc}$</td>
<td>Minimum time required to execute a quintic-polynomial trajectory given the maximum velocity and acceleration limits, respectively.</td>
</tr>
<tr>
<td>$V_i$, $V_T$</td>
<td>Interceptor’s and target’s velocity.</td>
</tr>
<tr>
<td>$V_{lim}$</td>
<td>The maximum robot end-effector velocity.</td>
</tr>
<tr>
<td>$V_{lim_\phi}$, $V_{lim_r}$</td>
<td>The maximum robot end-effector velocity divided along the $e_r$ and $e_\phi$ directions, respectively.</td>
</tr>
<tr>
<td>$v_x$, $v_y$</td>
<td>Target’s $x$ and $y$ Cartesian velocity, respectively.</td>
</tr>
<tr>
<td>$X(t)$</td>
<td>Continuous-time state vector.</td>
</tr>
<tr>
<td>$X_k$</td>
<td>State vector at time $k$.</td>
</tr>
<tr>
<td>$\hat{X}_{k+1,k}$</td>
<td>Predicted expected value of $X$ at time $(k+1)$ based on system at $k$.</td>
</tr>
<tr>
<td>$\hat{X}_{kk}$</td>
<td>Updated expected value of state (i.e., optimal estimate) at time $k$ based on system at $k$.</td>
</tr>
<tr>
<td>$X_R$</td>
<td>The robot’s position vector with respect to the Cartesian coordinate system.</td>
</tr>
<tr>
<td>$rX_R$, $\phi X_R$</td>
<td>The robot’s position components along the $e_r$ and $e_\phi$ directions, respectively.</td>
</tr>
</tbody>
</table>
$X_T$ The target's position vector with respect to the Cartesian coordinate system.

$\dot{X}_T, \ddot{X}_T$ The target's position components along the $e_r$ and $e_\phi$ directions, respectively.

$Y_k$ The noisy observed measurement matrix.

**GREEK LETTERS**

$\alpha$ Percentage of the maximum available torque used when upgrading the $a_{IPNG}$.

$\alpha_v, \alpha_a$ Percentages of the end-effector's velocity and acceleration limits, respectively, used when upgrading the $a_{IPNG}$.

$\alpha_s$ A smoothing constant used in recursive exponential smoothing.

$\beta$ Magnitude of the acceleration component added to the $a_{IPNG}$ in the LOS direction.

$\beta_v, \beta_a$ Values for $\beta$ such that the end-effector's velocity and acceleration, respectively, are maintained within desired percentages of their maximum values when upgrading the $a_{IPNG}$.

$\lambda$ Navigation gain.

$\theta_{LOS}$ The angle between the LOS and a reference line.

$\Phi$ State-transition matrix relating the object's past state to its current state.

$\tau$ Decorrelation constant, a parameter used by FOGM KF to adjust for time correlatability of target data.

$\sigma_x, \sigma_y$ Target's motion variance in $x$ and $y$ respectively.

$\Delta t$ Motion interval duration, describing the rate at which end-effector setpoints are sent to the robot's controller.
ACRONYMS

APPE Adaptive Prediction, Planning and Execution.
ARMAX Auto Regressive Moving Average with Auxiliary Inputs.
CCD Charged Couple Device.
CIMLab Computer Integrated Manufacturing Laboratory.
CPU Central Processing Unit.
CT Computed Torque.
EKF Extended Kalman Filter.
FOGM First-Order Gauss-Markov.
FMF Fading Memory Factor.
IPNG Ideal Proportional Navigation Guidance.
KF Kalman Filter.
LOS Line-of-Sight.
NC Numerically Controlled.
PC Personal Computer.
PD Proportional and Derivative.
PNG Proportional Navigation Guidance.
PPE Prediction, Planning and Execution.
PPNG Pure Proportional Navigation Guidance.
QPT Quintic-Polynomial Tracking.
Chapter 1: Introduction

1.1 Motivation

The onset of robotics in manufacturing environments has been a key factor in the move from hard automation to soft automation. The repeatability and ease of reprogramming makes robotic manipulation a suitable replacement to traditional pneumatic-based and human manipulation. To date, however, robotics has been predominantly used in applications where the environment can be engineered to suit the robot through the use of specialized part feeding devices and work-holding fixtures: robots have had much less impact in applications where the manufacturing environment and object placement cannot be accurately controlled.

Thus, in the typical application, the robot is “taught” off-line and is then programmed to repeat the task. End-position and proximity sensors are normally utilized to ensure that the current task has been successfully completed, and the subsequent operation can be initiated.

The move towards flexible automation in the manufacturing environment requires the industrial robot to be capable of performing tasks in a more autonomous manner. A typical manufacturing scenario is the picking of randomly-placed parts from a conveyor system or an automatically-guided vehicle whose motion is not known a priori. (Namely, parts need not be present in specific locations at specific times). In such unstructured environments utilization of vision sensors in conjunction with suitable motion-planning algorithms is a necessity, (e.g., [1], [2]).

The above application of industrial robots is the general area under study in this thesis; namely, the robotic interception of moving objects. Specifically, the navigation-guidance-based motion-planning scheme, developed by Mehrandezh et al. [3], to bring the end-effector of a robotic manipulator to a pre-grasping position matching both position and velocity of a target, will be used for on-line implementation using an industrial robot.
1.2 Literature Review

Considerable work has been done in the area of robotic interception of moving objects. If the target path is completely known a priori, time-optimal solutions can be computed. In [4], Park and Lee used optimal control theory to track a part moving on a conveyor. In [5], Zheng and Moore proposed a search method to determine the optimal interception point on a known target trajectory by a two-wheeled driven cart.

For autonomous motion-planning, however, the robot must be capable of intercepting a moving target without a priori information about its future trajectory. This problem has been investigated considerably during the past two decades, and various strategies have been proposed to solve the problem. These can be broadly classified into prediction-based techniques, visual-servoing techniques and navigation-based systems. A survey of the work carried out in each of these approaches is given in the following three sub-sections. Additionally, in Section 1.2.4, a survey of the experimental test-beds used to verify proposed interception methodologies is given.

1.2.1 Prediction-Based Techniques

If the object’s motion through the robot’s workspace can be accurately predicted, Prediction, Planning and Execution (PPE) systems can be used for object interception. PPE strategies involve three stages: predicting the target’s motion in the robot’s workspace, planning the robot’s motion to an anticipated rendezvous point on the predicted target trajectory, and execution of the planned motion.

Allen et al. [6] have used a PPE approach to intercept and grasp a moving toy train moving along an elliptical trajectory. An α-β-γ filter was used to track the train and provide target path prediction for robot trajectory planning. Okhotsimsky et al. [7] have proposed a PPE system to grasp a rod oscillating on a bifilar suspension. Data collected by the vision system during the first 0.5s is used to generate a model of the rod’s vibration (taking into account the dynamics of oscillations). This is used in conjunction with trapezoidal velocity-profile trajectories for the manipulator to plan the optimal capture point. The trajectory to the selected capture point is then executed.
Kimura et al. [8] used a PPE system to catch a free-flying ball. Target-motion is modeled by a parabola, using least-squares minimization to fit current target data to the model. Task-space, third-order polynomials are used to plan the manipulator motion to the rendezvous-point. A PPE strategy was also used to develop a Hanetsuki-playing robot (i.e., Japanese badminton) in [9].

The stages of PPE can be used in an "active" mode to ensure the successful completion of the interception task, hence APPE. The cornerstone of APPE techniques is the capability of re-planning the rendezvous-point and, thus, the robot trajectory in response to gross changes in the predicted target path. An APPE approach was used by Andersson [10] in the development of a robot ping-pong player, and by Buttazzo et al. [11] to catch a planar toy-mouse. In both these applications, the choice of the rendezvous-point was restricted to the intersection of the predicted object trajectory with pre-defined planes. Quintic polynomials were subsequently used to plan the robot's motion.

More recently, Fernandes and Lima [12] proposed an APPE system to catch ping-pong balls rolling on a table. In this system, target-motion prediction is predicted by fitting a quadratic function to the observed data. Object catching is performed at the edge of the table. This constraint defines the catching-point as the intersection of the predicted target trajectory with the catching line. (Velocity matching was not considered as an issue in this study.)

In all of the above mentioned applications, the planner can modify the rendezvous point as new target-motion data becomes available. However, the choice of the rendezvous point is limited to a set of candidate potential interception points corresponding to the intersection of the object's predicted trajectory with fixed planes or lines. A different approach was proposed by Croft et al. [13] to ensure time-optimal solutions to the moving-object-interception problem. In this APPE approach, the potential rendezvous points are not restricted to a small set of candidate points, but are selected anywhere along the predicted target's trajectory given the robot's motion constraints, Figure 1.1. The aim is to achieve globally-optimal interception of the moving target.
Figure 1.1: (a) Planning of time-optimal robot trajectories in APPE, and (b) Selection of optimal interception point using a travel / arrival time graph in APPE.

1.2.2 Visual-Servoing Techniques

Visual-servoing systems have been widely proposed to track and intercept moving objects. The objective function minimizes the difference in states of the target and the end-effector. In such systems, visual-data is utilized in the computation of the feedback signal. Features extracted from the images are used in conjunction with known camera models to estimate the target state with respect to a world coordinate system. Feedback is then computed by reducing errors in task space. This constitutes position-based servoing, [14]. Alternatively, in image-based servoing systems, feedback control values are computed directly from feature
vectors in the camera image plane (e.g., [15]). Recently, Malis and Chaumette [16] introduced the first hybrid position-based and image-based servoing system, referred to as the “2.5D approach”.

Acquiring and processing visual data causes an inherent delay in visual-feedback tracking systems. Prediction is commonly used to compensate for such delays. Koivo and Houshangi [17] used an auto-regressive prediction model to catch a cylindrical object moving in a plane. In their experiments, the vision sampling period was seven times longer than the control sampling period. Westmore and Wilson [18] used an Extended Kalman Filter (EKF) to track a moving object. Position-based servoing approach was utilized in both cases.

Feddema and Mitchell [19] proposed a feature-based trajectory generator for non-uniform sampling periods of the vision system. They used their technique to track a carburetor gasket moving on a circular parts feeder. Image-based visual feedback was used in an outer loop, while maintaining the robot’s inner feedback loop to overcome problems due to low visual sampling rate. More recently, Kelly et al. [20] used a two-loop visual controller, in which an image-based outer loop is used to compute the desired joint velocities and accelerations. These are in turn used in an inner velocity loop (based on the robot’s inverse dynamics) to drive the robot via the joint torques. They employed the two-loop direct visual controller to track a moving target by a planar direct-drive robotic arm. The arm moves in open loop with respect to image errors between two consecutive image samples (since the sampling rate of the velocity controller was 2.5 ms, whereas that of the visual feedback loop was 50 ms).

As stated above, in a visual-servoing system, the controller tries to match the current position and velocity of the object. Trying to match the end-effector velocity when the target is far is not advantageous, since it will slow down the robot. Lin et al. [21] addressed this problem by initially using a heuristic coarse-tuning method to bring the end-effector to the vicinity of the moving object. When the robot is within a pre-defined distance of the target, control is switched to a fine-tuning method with the objective of matching the target’s position and velocity. Their approach has parallels to the interception strategy under study in this
thesis, although herein an optimal selection of the switching point is performed on-line (as shall be explained later in Section 2.4).

A similar approach was taken in [22], where an on-line polynomial trajectory planner was used to drive a robot to the vicinity of a moving cube, and then switching to a position-based controller. In this paper, Lei et al. compared the use of position-based and image-based tracking schema, and concluded that the schema are equivalent, if the motion of the object is accurately estimated. Due to the initially large positional error, Ghosh et al. [23] added to the tracking plan an error-reduction term which is gradually reduced to zero. They used a multi-sensor fusion approach in an uncalibrated environment to track and pick a part moving on a rotating conveyor.

1.2.3 Navigation-Based Techniques

Navigation-based techniques embody a range of methods developed to intercept fast-maneuvering targets. They are predominantly used to track and intercept airborne missiles and evasive aircrafts. Interception normally refers to closing the distance between the interceptor and the target by bringing the former into a collision course with the latter. At interception, a large relative velocity is desirable.

Navigation-guidance techniques have been classified into five main categories, [24]: Line-of-Sight guidance, LOS, in which the interceptor is guided on an LOS course in an attempt to remain on a line joining the target and the point of control; Pursuit guidance, in which the interceptor continually points at the target; Proportional Navigation Guidance (PNG), which seeks to nullify the LOS rate while closing the distance; Optimal linear guidance, based on optimal control theory; and other guidance laws. Due to its simplicity of onboard implementation, PNG has been the most widely researched and commonly used guidance law (e.g., [25]).

There have been very few attempts to use guidance laws for robotic-interception of moving objects. The utilization of a navigation-based technique in robotics was first reported by Piccardo and Hondred [26] who utilized a PNG law to intercept a target moving on a straight line. No experimental results were reported. More recently, Su and Xi [27] have
proposed a new path-planning strategy for moving-object interception, called variant proportional guidance. In this method, which stems out from PNG, the angle between the LOS and the target velocity is changed at a constant rate. The method enables the grasping pose of the gripper to be automatically adjusted during the intercept course. In their experimental work, both maneuvering and non-maneuvering targets were successfully intercepted.

In both [26] and [27], final-velocity matching was not presented as an issue. In contrast, this issue was addressed in the method proposed by Mehrandezh et al. [3]. This navigation-guidance-based technique, which constitutes the basis of this thesis, has been developed to intercept randomly moving targets, making full use of the robot's capabilities. It leads to fast interception, without requiring target future-path prediction. Its main advantage is the simplicity of on-line computation. It is essentially a hybrid method, in which a tracking method takes over control of the robot from the navigation-guidance method at an optimal instant to ensure that the robot matches both target position and velocity at interception. An overview of the method will be given in Chapter 2.

1.2.4 Implementation Issues in Robotic Moving-Object Interception

The main requirements for a robotic moving-object interception system are vision sensing, trajectory planning and robot-arm control. For successful interception, real-time coordination between these systems must be provided. In the previous sections, different strategies for using sensory information to generate trajectory plans were reviewed. In this section, systems built to test different moving-object interception algorithms in real-time will be examined in terms of the components utilized and their integration.1

The configuration of practically all systems reviewed is a multi-architectural system, in which the different functions of image acquisition and image processing, planning and arm control are separated into individual computers such that they can be performed in parallel to ease the computational burden and hence processing delays. The approach of running these three functions on three separate systems was used by Papanikolopoulos et al. [30] in their

---

1 Corke provides an exhaustive review of implementation issues pertaining to general vision-based robotic systems in [28], and more recently in [29].
experiments on vision-guided grasping of moving objects and by Hujic et al. [31] in their experimental APPE based system.

In some cases, the vision system and the planner were integrated in one computer. This minimizes communication delays and avoids some synchronization problems, although the computational burden on the main processor is increased, e.g., [11], [12]. A common platform is the PC (e.g., [7], [12]), although SUN host systems have also been commonly utilized (e.g., [8], [30]). Since functions run independently of one another, combination of platforms is not uncommon, e.g., an SGI workstation planner linked to an IBM PC image processor in [23]. A rare use of a Mackintosh in a real-time environment was chosen by Sharma et al. [32] in their PPE system. In the great majority of cases a C compiler has been utilized for real-time experimentation, although in [7] and [33] some algorithms were implemented using Pascal and Matlab, respectively.

By far, the most commonly used robotic-manipulator is the Unimate PUMA 560. Its controller has an ‘ALTER’ facility, allowing trajectory modifications to be received via a serial communications channel, at a sampling rate of 28 ms. Custom-built manipulators have also been used. For example, Kelly et al. [20] have used a planar, two degree-of-freedom direct-drive manipulator designed and built at CICESE Research Center, in Mexico. Kimura et al. [8] used a four degree-of-freedom manipulator designed and built at MIT and Papanikolopoulos et al. [34] used the Direct-Drive Arm II built at Carnegie Mellon University. Other robotic systems have also been used, such as the GMFanuc S-100 in [31] and the Mitsubishi Movemaster II in [32]. A two degree-of-freedom reconfigurable robot (assembled from a Robix Robot Construction Set) was utilized by Piepmeier et al. [33] in their moving-target tracking experiments in uncalibrated environments.

In the moving-object interception experiments, the fixed camera is the most common set-up, e.g., [20], [31]. Eye-in-hand configurations have also been utilized, e.g., [30], although to a much lesser extent. Additional to vision sensors, a rare use of a laser rangefinder was reported in [35].
Since the experiments are concerned with real-time visual control for end-effector coordination, rather than scene understanding, image processing is generally simplified by working in structured environments where a white object moves on a dark background, e.g., [31], [33]. A priori knowledge of the kinematic robot model and the camera calibration model are usually utilized in the experiments. However, more recently, research in target tracking and catching in uncalibrated environments has been conducted (within image-based visual servo control), e.g., for moving target tracking in [36], and for part picking from a rotary conveyor in [23].

1.3 Research Objective

The primary objective of this thesis is the implementation and testing of the IPNG-based interception technique developed by Mehrandezh et al. [3] via an industrial set-up available in the CIMLab. Through this implementation, the viability of a navigation-guidance-based method (versus conventional visual-servoing techniques) will be illustrated for robotic-interception of moving objects in semi-structured environments.

The above mentioned hybrid IPNG-based interception technique has two primary (sequential) on-line robot-motion planning phases: IPNG, whose objective is to bring the end-effector close to the target as fast as possible, and tracking, whose objective is to match the target's position and velocity. (Switching from one phase to the next needs to be performed in an optimal manner).

One must note, however, that the motion-planning algorithms proposed in [3] were developed for the ideal case where the robot dynamic model is available. This is not normally the case for industrial robots, whose kinematic and dynamic models are usually unknown, and, furthermore, whose controllers are closed to the user. Thus, for the implementation of the IPNG-based interception scheme in the CIMLab, new motion-planning algorithms that can servo the industrial robot without these models and with no direct access to the (joint) motion controllers had to be developed.

Within the above framework, the following tasks were set in order to achieve the overall implementation-methodology-development objective of this thesis:
(1) Development of an overall implementation strategy:
   (1.1) Identify hardware and software architectures;
   (1.2) Identify necessary modifications to the interception-planning algorithms - robot-
          motion planning and target tracking.
(2) Development of robot-motion planning schema:
   (2.1) Modify the IPNG-based robot-motion planning algorithms to generate end-effector
          motion commands using velocity and acceleration limits (versus torque limits
          originally used in [3]) - for Phase I;
   (2.2) Develop a new tracking algorithm based on quintic polynomials in task space (as
          opposed to Computed Torque (CT) tracking originally used in [3]) - for Phase II;
   (2.3) Develop a new (optimal) phase-switching algorithm for the motion-planning
          algorithms developed within the framework of this thesis (Tasks 2.1 and 2.2. above).
(3) Development and testing of an experimental system:
   (3.1) Develop (PC-based) robot-motion planning software modules;
   (3.2) Develop an image-acquisition software module for moving-target tracking (including
          camera calibration);
   (3.3) Establish all software communication linkages (including one to the CIMLab’s
          Kalman-Filter-based target-motion prediction software module);
   (3.4) Establish all hardware interfaces between PCs, robot controller and vision system;
   (3.5) Carry out moving-object interception tests and analyze the results.

1.4 Implementation-Strategy Overview and Thesis Organization

A schematic diagram of the developed IPNG moving-object interception system is
illustrated in Figure 1.2.

The vision module provides real-time target tracking. The target consists of a
premarked object displaced on a numerically controlled xy-table. The image coordinates of the
object’s centroid are converted into the robot’s world coordinates using a camera-calibration
model computed off-line. The ‘raw’ target positional data is fed into a prediction module
(based on a Kalman filter) to compensate for system delays and sensor noise, and to provide
one-step ahead target state prediction (necessary during the tracking phase of interception). The motion-planning module uses the information about the robot and target states to generate the robot's motion. Finally, the planned motion is executed by the manipulator.

Figure 1.2: IPNG-based moving-object-interception system.
A review of the IPNG robotic interception scheme originally developed by Mehrandezh et al. [3] is given in Chapter 2. Modifications to the original method are also presented in this chapter, supported by simulation results. Chapter 3 discusses the limitations of the robot controller at the CIMLab and subsequently the development of the implemented new motion-planning algorithms. The prediction module is also discussed in Chapter 3.

The IPNG-based moving-object interceptor test-bed is presented in Chapter 4. Various experimental issues pertaining to individual components and system integration are discussed and experimental results are presented. In Chapter 5, the tracking method utilized in this thesis is modified to perform target tracking in a similar manner to IPNG. This new tracking method is proposed and examined to provide a basis for recommendations for future work in this area.

Finally, the conclusions of the thesis are presented in Chapter 6, which also discusses recommendations for future work.
Chapter 2: Motion-planning

2.1 Introduction

In this chapter, the development of an IPNG-based robotic interception scheme is outlined. An overview of Proportional Navigation Guidance interception systems for airborne interception is described first, followed by a description of the IPNG scheme. A review of the IPNG-based robotic interception method implemented in this thesis is presented in Section 2.4. In particular, three issues are addressed: robot-motion initialization, limiting or upgrading the acceleration command, and switching to a conventional tracking method ensuring a smooth grasp.

Although this chapter comprises primarily background material, a new motion-initialization scheme developed within the framework of this thesis is described in Section 2.4.2. Simulation examples are presented at the end of the chapter, to demonstrate the validity of the IPNG-based robotic-interception scheme and of the proposed contribution.

2.2 Proportional Navigation Guidance

Amongst the five major classes of guidance laws mentioned in Chapter 1, the Proportional Navigation Guidance (PNG) scheme has attracted a considerable amount of interest in the missile-guidance literature. PNG laws seek to nullify the LOS rate, while closing on the target. (PNG has its origins among the ancient seafarers, who realized that if two constant-velocity vessels maintained a constant relative bearing while closing their relative distance, collision was ensured).

PNG was introduced in the 1940s and has been widely used for the interception of airborne targets due to its simplicity of on-board implementation. It has demonstrated acceptable performance and success in a large number of fielded missile weapon systems, [24]. There exist two major categories to PNG, [25]: LOS-referenced class and velocity-referenced class. The former is superior in terms of mathematical tractability (i.e., less
sensitivity to the initial conditions of the interception), whereas the latter is more practically implementable in aerodynamic environments.

2.2.1 Target versus Interceptor Motion Geometry

Figure 2.1 depicts the intercept geometry of an interceptor in planar pursuit of a target, with the coordinate frame's origin located at the interceptor. In guidance-law formulation, the interceptor and the target are considered as point masses. The position vector of the target relative to the Cartesian coordinate frame is assumed to be known.

It is more convenient to formulate LOS-referenced guidance schema in terms of a polar coordinate system located at the interceptor, [37]. The unit vectors of this coordinate frame are denoted as \( e_r \) and \( e_\theta \), where \( e_r \) is in direction of the LOS. \( \mathbf{r} \) is the relative position vector between the interceptor and the target, and \( \theta_{LOS} \) is the angle which the LOS makes with an inertial reference line. \( V_T \) denotes the target's current velocity, and \( V_I \) denotes the interceptor's velocity.

![Figure 2.1: Interception geometry.](image)
2.2.2 Proportional Navigation Guidance Schema

Yang and Yang [25] developed a unified scheme under which PNG systems can be described. In this generalized expression, the commanded acceleration of the interceptor, \( a_{\text{PNG}} \), is proportional to the LOS:

\[
a_{\text{PNG}} = \lambda L \times \dot{\theta}_{\text{LOS}},
\]

(2.1)

where \( \lambda \) is the navigation gain, \( \dot{\theta}_{\text{LOS}} \) is the angular velocity of the LOS and \( L \) is the normal direction of the acceleration command. Different PNG schema employ different forms of \( L \). In True Proportional Navigation Guidance (TPNG), (which belongs to LOS-referenced PNG class), \( L \) is along the LOS (\( L = \hat{r}_o e_r \)). In Pure Proportional Navigation Guidance (PPNG), \( L \) is along the interceptor's velocity, i.e., \( L = -V_i \). PPNG is a velocity-referenced scheme.

Ideal Proportional Navigation Guidance (IPNG) is a LOS-referenced system. In IPNG, \( L \) is computed as the relative velocity vector, \( \hat{r} \):

\[
L = \hat{r}e_r + r\dot{\theta}_\theta e_\theta.
\]

(2.2)

In airborne interception systems, IPNG does not have much practical use compared to PPN. However, its robustness to the initial conditions of the interceptor and its more relaxed capture criterion makes it very suitable for the robotic interception of moving objects, [3].

2.3 Ideal Proportional Navigation Guidance

The IPNG law has been developed by Yuan and Chern [38]. In this scheme, the acceleration command tries to turn the relative velocity to the direction of the LOS with utmost effort. The IPNG acceleration command is applied in the direction normal to the relative velocity between the interceptor and its target, and its magnitude is proportional to the product of the LOS rate and the relative velocity:

\[
a_{\text{IPNG}} = \lambda \hat{r} \times \dot{\theta}_{\text{LOS}}.
\]

(2.3)

\( \dot{\theta}_{\text{LOS}} \) can be expressed as:

\[
\dot{\theta}_{\text{LOS}} = \frac{r \times \hat{r}}{|r|^2}.
\]

(2.4)
By substituting Equation (2.4) into Equation (2.3), the IPNG acceleration command can be computed as,

\[ a_{IPNG} = \frac{\lambda}{|r|^2} \{ \mathbf{r} \times (\mathbf{r} \times \dot{\mathbf{r}}) \}. \] 

(2.5)

Since \( a_{IPNG} \) is perpendicular to the relative velocity, the magnitude of the relative velocity is maintained as constant and its direction is turned onto the LOS during the intercept period.

In [38], it was proven that for IPNG, the capture criterion is \( \lambda > 1 \), no matter what the initial condition of \( \dot{\mathbf{r}} \) and target maneuver are. This implies that for both maneuvering and non-maneuvering targets, successful interception is achieved provided that \( \lambda > 1 \). Moreover, for non-maneuvering targets, \( \dot{\Theta}_{LOS} \) approaches zero, when \( \lambda > 2 \). From Equation (2.4), it can be seen that if \( \dot{\Theta}_{LOS} = 0 \), then, the relative velocity lies along the LOS. This implies that the interceptor is locked onto the target, and the dimensionality of the interception problem is reduced to one (i.e., along \( e_r \)), where the interceptor has to close in on the target.

Consequently, \( \lambda \) should be chosen to be greater than 2, for successful interception.

2.4 Robotic Interception via Ideal Proportional Navigation Guidance, [3]

Mehrandezh et al. [3] have adapted the IPNG scheme for robotic interception. Unlike conventional tracking methods, which continually minimize the difference between the states of the robot and the target, IPNG tries to nullify the time-rate of change of the LOS angle. During interception, the relative velocity is turned onto the LOS, thus, increasing the closing-velocity of the robot towards the target. This is a major advantage over tracking methods which may decrease the velocity of the robot even when the latter is far from the target.

IPNG was designed to bring the interceptor on a collision course with the target. However, for robotic applications, a smooth grasp is necessary, whereby the end-effector matches both the location and the velocity of the target. For robotic interception of moving objects, the IPNG technique, therefore, must be combined with a conventional tracking method. Thus, in the hybrid IPNG-based interception scheme, motion-planning is divided into
two phases: Phase I, during which the robot is under IPNG control, and Phase II, during which the robot is controlled using a conventional tracking method, Figure 2.2.

2.4.1 Robot Dynamics

The motion of an n degree-of-freedom robotic manipulator is governed by the dynamic equation, [39],

\[ M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = T, \]  

(2.6)

where \( q \in R^n \) is the vector of joint angles, \( T \in R^n \) is the torque vector, \( M(q) \in R^{n \times n} \) is the inertia matrix, \( C(q, \dot{q})\dot{q} \in R^n \) is the Coriolis and centripetal force vector, and \( G(q) \in R^n \) is the torque vector due to the gravitational force. The torque vector, \( T \), is subject to dynamic constraints as:
where $T_{i\text{max}}$ is the maximum torque available in the $i^{th}$ actuator. Equation (2.6) can be expressed in task space as:

$$MJ^{-1}\{\ddot{X}_R - \dot{J}^{-1}\dot{X}_R\} + CJ^{-1}\dot{X}_R + G = T,$$  \hspace{1cm} (2.8)

by using mappings between the joint coordinates $q$ and the robot end-effector coordinates $X_R$:

$$X_R = P(q),$$ \hspace{1cm} (2.9)

$$\dot{X}_R = J(q)\dot{q},$$ \hspace{1cm} (2.10)

$$\ddot{X}_R = \dot{J}(q)\dot{q} + J(q)\ddot{q}.$$ \hspace{1cm} (2.11)

$P(q)$ represents the forward kinematic relation for the end-effector and $J(q)$ is the end-effector Jacobian matrix. (The arguments of $M$, $C$ and $G$ in Equation (2.8) have been dropped for simplicity). In Equation (2.8), $\ddot{X}_R$ can be replaced with $a_{\text{IPNG}}$, which is computed as described in Section 2.3. Upon rearranging, the dynamic equation of motion can be re-written as:

$$MJ^{-1}a_{\text{IPNG}} + \left\{C - MJ^{-1}\dot{J}\right\}J^{-1}\dot{X}_R + G = T.$$ \hspace{1cm} (2.12)

Equation (2.12) is the dynamic equation of motion when utilizing the IPNG technique, [3].

From Equation (2.5), it can be seen that, if the manipulator is initially at rest, the initial acceleration command will be in a direction perpendicular to the target’s velocity. This may clearly be non-optimal. Thus, a motion initialization scheme is required in the beginning of the interception. This issue is discussed in Section 2.4.2.

During the IPNG-phase of interception, the computed acceleration command can be modified, to reflect the maneuvering superiority of a robotic manipulator when compared to an airborne interceptor. This issue will be addressed in the Section 2.4.3. Subsequently, in Section 2.4.4, the tracking phase of the hybrid motion-planning technique is discussed.
2.4.2 Robot-Motion Initialization

In [3], it was proposed to initialize the robot motion by sending the robot towards the current position of the target, (i.e., along the LOS direction), using the maximum permissible acceleration. Robot motion-initialization was terminated when either the robot had reached its maximum permissible velocity or when a pre-defined time threshold was passed.

The direction of robot-motion initialization clearly influences the direction of the relative velocity when IPNG takes over control. As discussed in Section 2.3, IPNG tries to turn the relative velocity onto the LOS with utmost effort, to achieve a constant heading with the target while closing the distance. The strategy of initiating the robot motion along the LOS aims at initializing the robot with a closing velocity; however, no effort is made to simultaneously initialize the robot with a heading velocity perpendicular to the LOS (to try to match the target’s velocity component in this direction).

Therefore, in this thesis, it is proposed to initialize the robot in a direction which takes into account the direction of the target’s current velocity, as:

\[ e_d = \frac{ke_r + e_{xt}}{|ke_r + e_{xt}|}, \tag{2.13} \]

where \( e_r \) is a unit vector along the LOS, \( e_{xt} \) is a unit vector along the target’s velocity and \( k \) is a pre-defined coefficient, \( k > 1 \). The selection of \( k > 1 \) prevents \( e_d \) from collapsing to zero, if the direction of the target’s velocity is directly opposite to \( e_r \), and ensures that the robot will still be initiated towards the target’s location in such a case. A more conservative approach could be the use of a higher value of \( k \), \((k > 2)\), to place more emphasis on initializing the robot towards the current location of the target.

If the target’s velocity is not available in the first iteration, the robot is initialized along the LOS for the first control sampling period. Subsequently, if the relative velocity is turned onto the LOS, (i.e., if \( |\dot{\theta}_{LOS}| < \text{threshold} \)), the direction of motion initialization is set as:

\[ e_d = e_r, \tag{2.14} \]

(i.e., along the LOS), for the remaining duration of robot-motion initialization.
2.4.3 Modifying the Acceleration Command

IPNG was originally developed for airborne interception, where the interceptor and the target are normally assumed to be capable of maneuvering in a direction normal to their current direction of motion. However, the end effector of a robotic manipulator can be maneuvered in any direction, given that the dynamic constraints of the actuators are not violated. To reflect this superiority, the acceleration command of the robotic interceptor should be upgraded while maintaining the torques within a pre-defined percentage, \( \alpha \), of their maximum values (for \( 0 < \alpha < 1 \)). The acceleration command was boosted in [3] by adding a component in LOS direction, i.e.,

\[
a_c = a_{IPNG} + \beta e_r, \tag{2.15}
\]

where \( \beta \) is a constant that is computed on-line, such that,

\[
| T_i | \leq \alpha | T_{lim} |, \ i=1,...,n. \tag{2.16}
\]

It should be noted that, the effect of upgrading the acceleration command along the LOS is to increase the closing velocity, making the robot close in on the target faster.

Conversely, if the torques computed in Equation (2.12) violate the limits (defined by Equation (2.7)), the acceleration command must be restricted. The acceleration command can be limited by scaling \( a_{IPNG} \), i.e., changing its magnitude while keeping the original direction of acceleration,

\[
a_c = K a_{IPNG} - \tag{2.17}
\]

In Equation (2.17) above, the coefficient \( K \) is determined on-line such that, given the current robot configuration, none of the actuator limits is exceeded.

The algorithm for modifying the IPNG acceleration command is depicted in Figure 2.3.

2.4.4 Hybrid Interception Method

As mentioned earlier, navigation-based techniques are designed to bring the interceptor into a collision course with a moving target. At interception, a large relative velocity is desirable. However, for robotic applications in manufacturing settings, a smooth grasp is necessary. Therefore, robotic interception is defined as,
Figure 2.3: Modifying the computed acceleration command, (adapted from [3]).

\[
|p| < tol_p, \quad (2.18a)
\]

and

\[
|\dot{p}| < tol_v, \quad (2.18b)
\]

where \(tol_p\) and \(tol_v\) are user-defined tolerances for relative position and velocity errors at interception.

Clearly, the IPNG scheme does not meet the second requirement of minimal velocity error, and, so, control of the robot must be switched to a tracking method which brings the manipulator's end-effector into a pre-grasp situation satisfying Equations (2.18a) and (2.18b) simultaneously. The proposed interception scheme, therefore, is a hybrid method in which IPNG-based motion-planning first brings the robot to the vicinity of the target as fast as possible, and then, a tracking-method takes control to slow down the robot for a smooth interception.

The overall interception time, \(t_{int}\), is given by,
\[ t_{\text{int}} = t_{\text{IPNG}} + t_{\text{Tracking}}, \]  

(2.19)

where \( t_{\text{IPNG}} \) and \( t_{\text{Tracking}} \) denote the times during which the robot is under IPNG control and tracking control, respectively. (The time during which the robot is being initialized is hereby included in \( t_{\text{IPNG}} \)). The switching time, \( t_{\text{sw}} \), is equal to \( t_{\text{IPNG}} \).

Since the interception scheme comprises two different methods, the interception time depends on the switching instant. For time-optimal interception, the tracking method must take over control of robot-motion planning at an optimal time instant. Therefore, while the robot is under IPNG control, the decision whether to switch to tracking control or proceed with IPNG must be considered at each planning instant.

In [3], Mehrandezh et al. utilized a PD type, Computed-Torque (CT) tracking method, [40]. In this tracking technique, the torque is computed via the inverse dynamics of the manipulator. Mehrandezh et al. showed that if the robot's dynamic model is known exactly, the time required for the CT tracking method to reduce the relative position and velocity errors to zero can be estimated precisely on-line, using the current states of the robot and the target. This estimate, \( \tilde{t}_{\text{CT}} \), is used to predict the overall interception time, \( \tilde{t}_{\text{int}} \), if control is switched to tracking at that instant, using:

\[ \tilde{t}_{\text{int}} = t_{\text{IPNG}} + \tilde{t}_{\text{CT}}. \]  

(2.20)

This estimate is used to determine the optimal switching instant on-line.

### 2.5 Computer Simulations

A test-bed for simulating the IPNG robotic interception scheme on Matlab has been developed in the CIMLab, [3]. A two degree-of-freedom planar manipulator was modeled, making use of a Robotic Toolbox for Matlab, [41]. The physical details of the manipulator are depicted in Figure 2.4. Target-motion is simulated using pre-defined trajectories. The target is assumed to be a point mass moving in the same plane of the manipulator, and it is assumed that the target's instantaneous location is available through a vision system.

In the simulations, the robot is initially at rest with the end-effector located at \( [0 \ 1]^T \) m in the xy-plane. The tolerances of interception are defined as: \( tol_p = 15 \) mm and \( tol_v = 15 \) mm.
mm/s. The coefficient $\alpha$ in Equation (2.16) is 0.5, and $k$ in Equation (2.13) is set as 3.0. The navigation constant is set at 3.0, and proportional and derivative gains of $K_p = 1.0I_{2\times2}$ and $K_d = 2.0I_{2\times2}$ are employed in the CT tracking method, [3].

![Schematic diagram of the simulated two-link manipulator.](image)

**Figure 2.4:** Schematic diagram of the simulated two-link manipulator.

Figure 2.5 shows the simulation results for a target moving on a sinusoidal path (i.e., fast-maneuvering motion). The target’s path, $X_T$, is defined as:

$$X_T = \begin{bmatrix} 1.0 + 0.2 \sin \left( \frac{\pi}{2} t \right) \\ 1.0 - 0.2 t \end{bmatrix} \text{m.} \quad (2.21)$$

Using IPNG, interception occurs at 4.0 s. The pure CT-based tracking method yields 6.65 s. (If one were to use IPNG with initialization towards the current target position, as proposed in [3], interception would occur at 4.2 s). Figure 2.6 depicts the variation of the $x$ and $y$ components.
of the position and velocity of the end-effector (using IPNG with the proposed initialization) and the target versus time.

![Graph showing interception methods](image)

**Figure 2.5**: Interception of a target moving on a sinusoidal path.

![Graphs showing position and velocity variations](image)

**Figure 2.6**: Position and velocity variation versus time for end-effector and target.
Figure 2.7 shows how the interception time varies with the switching time. The steep increase in interception time beyond the optimal switching point indicates that late switching will lead to a high penalty on the interception time. This increase is due to overshoot in the robot’s trajectory.

![Graph showing variation of interception time with switching time](image)

**Figure 2.7:** Variation of the interception time versus switching time for the example depicted in Figure 2.5.

An additional example is provided herein to further illustrate the advantage of the proposed initialization scheme versus Mehrandezh’s when intercepting a target moving on a parabolic path, Figure 2.8. The target’s trajectory is defined as:

\[
X_T = \begin{bmatrix} 0.2t \\ 0.1t - 0.05t^2 \end{bmatrix} \text{ m.} \tag{2.22}
\]

The interception time using the proposed initialization scheme is 3.15 s, whereas when using the original initialization, interception occurs at 3.4 s. Figure 2.9 depicts the variation of the \(x\) and \(y\) components of the position and velocity of the end-effector (using IPNG with the proposed initialization) and the target versus time.
Figure 2.8: Interception of a target moving on a parabolic path.

Figure 2.9: Position and velocity variation versus time for end-effector and target.
2.6 Summary

The IPNG-based interception system originally proposed in [3] for the robotic interception of moving objects has been presented. The scheme is essentially a hybrid method, in which a CT-tracking technique must take over control of the robot. The simulation results show that the hybrid method is superior to the pure CT tracking method. As proposed in this thesis, a further improvement in the interception time can be obtained by modifying the direction of robot motion initialization, by taking into account the target's velocity.

In the next chapter, the motion-planning algorithms presented in this chapter are modified to enable their implementation on the industrial robotic manipulator available in the CIMLab.
Chapter 3: Implementation of the Ideal-Proportional-Navigation-Guidance-Based Interceptor – Motion-Planning Issues

3.1 Introduction

The motion-planning algorithms presented in the Chapter 2 utilize the inverse dynamics of the robot to compute robot-motion commands. However, as is the case for most industrial robots, the dynamic model of the robotic manipulator in the CIMLab is not available. Thus, user control of the manipulator is achieved by sending positional data to the robot’s controller on-line.

The IPNG-based motion-planning algorithms originally developed in [3] were modified in this thesis for the effective control of the GMFanuc S-100 manipulator in the CIMLab. These algorithms are discussed in detail in this chapter. A description of the prediction system utilized by the interception system is also given. The chapter is concluded with some simulation examples, which provide a preamble to the actual experimental work presented in Chapter 4.

3.2 The GMFanuc S-100 Manipulator

A GMFanuc S-100 manipulator is utilized in the implementation of the IPNG-based interception technique at the CIMLab. The GMFanuc S-100 is a six degree-of-freedom manipulator, with primarily three revolute joints to govern end-effector translation, and three revolute joints mounted at the wrist for orientation. The robot is controlled by a Karel controller, which runs its own operating system, [42]. The Karel controller performs the joint-level control of the manipulator, to which the user does not have access.

Programs can be written in the Karel high-level language to control the operation of the robot’s end-effector. The controller allows serial communication with a host-computer during the execution of a program. This feature is used herein to instruct the robot to execute real-
time trajectories generated by the host-computer, by feeding the robot with task-space position data and specifying a segment of time during which the robot must reach the given position. The Karel controller will subsequently map the end-effector position from the task space coordinate system to joint space, and perform low-level control to achieve the desired motion. The controller performs joint-interpolated motion between consecutive task-space setpoints on-line. The dynamic model of the manipulator is not available. The performance of the manipulator is, therefore, constrained herein by conservative end-effector velocity and acceleration limits.

The problem of autonomous robot motion has been traditionally divided into two separate tasks: trajectory planning and robot control, whereby the latter is designed to reliably follow the trajectory planned by the former, e.g., [43]. In the implementation of the proposed IPNG-based interception method, a predictor is utilized to observe the object's motion and feed this information into the GMFanuc S-100's motion planner, Figure 3.1. The function of the planner is to compute the desired trajectory setpoints of the end-effector based on the IPNG method. The Karel controller utilizes these setpoints to produce the inputs to the manipulator so that the end-effector tracks the desired path. Within this framework, the algorithms developed in this thesis pertain entirely to robot-motion planning.

![Figure 3.1: Architectural organization of GMF S-100 control, (adapted from [44]).](image-url)
There exists a large body of literature in the field of time-optimal robot-trajectory planning (e.g., [45], [46]). While these and various other approaches not discussed herein usually yield optimal paths, the computation times required for finding optimal PTP solutions and the consideration of robot dynamics render them prohibitive for on-line use in the present system in the CIMLab.

One must recall that the motion-planning algorithms presented herein aim at bringing the robot to a pre-grasping position. Once the end-effector is at a pre-grasping position, it is assumed that a proximity-sensor-based tracker will take over control until the target is successfully grasped. Furthermore, only the translation of the end-effector is considered in this thesis, although, the algorithms can be extended to cope with end-effector orientation. Naturally, it is assumed, all throughout the thesis, that the robot has a clear velocity and acceleration superiority compared to the moving object.

3.3 Kalman Filter Prediction

In robotic-interception schema, target-motion is commonly observed via vision sensors. Latency effects due to vision delays and noise corruption inherent in the vision system necessitate smoothing of the information and short-term target-motion prediction. Digital filters are normally employed to estimate future target states based on the current and previous data. Common filters used in robotic-interception schema include ARMAX filters (e.g., [47], [48]), \(\alpha\)-\(\beta\)-\(\gamma\) filters (e.g., [49]) and Kalman Filters (e.g., [31], [50]). A comparison of these estimation approaches in target tracking can be found in [51].

The purpose of the prediction system within the hybrid IPNG-based interception scheme developed herein is to provide accurate target tracking and short-term target-trajectory prediction (typically in less than 0.7s). In our laboratory, extensive work has been carried out on the development of a KF-based target-motion-tracking and -prediction system, ([52], [53]). This prediction system has also been utilized in this thesis. A brief overview of the prediction model follows.
### 3.3.1 The Kalman Filter (KF) ([54], [55])

The KF is a computationally efficient recursive filter that generates an optimal least-squares estimate from a sequence of noisy observations. Let a discrete-time linear dynamic system be described by the system model:

\[ X_{k+1} = \Phi X_k + V_k, \quad V_k = N(0, Q_k) \]  

where \( X_k \) is the state vector, \( \Phi \) is the state transition matrix (which transitions the state vector \( X_k \) at time \( k \) to the state vector \( X_{k+1} \) at time \( k+1 \)) and \( V_k \) is a sequence of zero-mean white Gaussian noise with covariance \( Q \), representing the dynamic model driving noise vector. The observation model is given as:

\[ Y_k = M X_k + W_k, \quad W_k = N(0, R_k) \]

where \( Y_k \) is the measurement matrix, \( M \) is the observation matrix and \( W_k \) is a sequence of zero-mean white Gaussian noise with covariance \( R \), denoting the observation error.

The procedure for the recursive formulation of the KF is carried out in two stages: the prediction stage (Equations (3.3a) and (3.3b)) and the update stage (Equations (3.3c) through (3.3e)).

(i) **Predictor Equation** (one-step ahead predictor):\
\[ \hat{X}_{k+1,k} = \Phi \hat{X}_{k,k} \quad \] (3.3a)

(ii) **Predictor Covariance Matrix Equation:**\[ \hat{S}_{k+1,k} = \Phi \hat{S}_{k,k} \Phi^T + Q_k, \quad \] (3.3b)

where \( \hat{S}_{k+1,k} = \text{COV}(\hat{X}_{k+1,k}) \), giving a measure of the accuracy in predicting \( X \) at time \( k+1 \) based on the measurements made at time \( k \) and before.

(iii) **Kalman Gain Matrix Update:**\[ H_{k+1} = \hat{S}_{k+1,k} M^T \left[ R_{k+1} + M \hat{S}_{k+1,k} M^T \right]^{-1}. \] (3.3c)

The Kalman Gain, \( H_{k+1} \), is used to obtain the optimal estimate of the current state from new measurement data \( Y_{k+1} \) and the predicted state \( \hat{X}_{k+1,k} \).

(iv) **Filtering Equation:**\[ \hat{X}_{k+1,k+1} = \hat{X}_{k+1,k} + H_{k+1} (Y_{k+1} - M \hat{X}_{k+1,k}). \] (3.3d)
(v) **System Covariance Update:**

\[
\hat{S}_{k+1,k+1} = (I - H_{k+1}M)\hat{S}_{k+1,k}.
\]  

(3.3e)

In the CIMLab, to further improve the system's tracking performance, a Fading Memory Factor (FMF) was introduced in [52]. Through the FMF, old data is discarded by increasing the covariance of the measurement noise for past measurements, [54]. This was achieved by multiplying the error covariance by an age-weighting factor \( s, (s > 1) \) in Equation (3.3b):

\[
\hat{S}_{k+1,k} = s\Phi \hat{S}_{k,k} \Phi^T + Q_k.
\]

(3.4)

### 3.3.2 Object-Motion Estimation via Kalman Filter [52]

In the current system, target data obtained from the vision system are first converted into world coordinates using the camera calibration model proposed by Tsai, [56], and, then, processed by a linear KF. The target-motion model proposed by Singer, [57], was used. In this model, the target’s jerk is described as a First-Order Gauss-Markov (FOGM) process:

\[
\begin{align*}
\dot{p}_x &= v_x, \\
\dot{v}_x &= a_x, \\
\dot{a}_x &= \frac{1}{\tau}a_x + \frac{2}{\tau}\sigma_x V_x,
\end{align*}
\]

(3.5a)

\[
\begin{align*}
\dot{p}_y &= v_y, \\
\dot{v}_y &= a_y, \\
\dot{a}_y &= \frac{1}{\tau}a_y + \frac{2}{\tau}\sigma_y V_y,
\end{align*}
\]

(3.5b)

\[
\begin{align*}
\dot{p}_x &= v_x, \\
\dot{v}_x &= a_x, \\
\dot{a}_x &= \frac{1}{\tau}a_x + \frac{2}{\tau}\sigma_x V_x,
\end{align*}
\]

(3.5c)

where \( p, v \) and \( a \) are the target's planar position, velocity and acceleration, respectively. \( \tau \) is the acceleration decorrelation time (i.e., a measure of how quickly the object changes its trajectory) and \( V \) is Gaussian white noise with variance \( \sigma \). The continuous-time state vector is given by:

\[
X(t) = \begin{bmatrix}
p_x \\
v_x \\
a_x \\
p_y \\
v_y \\
a_y
\end{bmatrix}.
\]

(3.6)

By adjusting \( \sigma \) and \( \tau \), good tracking performance was obtained in [52].
3.4 Robot-Motion Planning

In this section, the motion-planning algorithms presented in Chapter 2 are modified to suit the Karel controller of the GMFanuc S-100 manipulator in the CIMLab. The Karel controller is servoed using end-effector setpoints. Thus, the scope of the robot-motion planning module on the experimental testbed is to use the IPNG-based technique to compute the end-effector position, in response to the target’s motion. The motion-planning algorithms are designed to make full utilization of the capabilities of the GMFanuc S-100 manipulator, constrained herein by conservative end-effector velocity and acceleration limits.

Motion-planning is divided into two main phases in the hybrid interception scheme: Phase I, during which the end-effector’s position is computed via the IPNG technique, and Phase II, during which the end-effector’s position is computed via a conventional-tracking technique, Figure 2.2. In Phase I, a motion-initialization scheme is utilized prior to IPNG-based control. During IPNG control, the acceleration command is modified subject to the specified velocity and acceleration limits. In Phase II of the hybrid scheme, a tracking technique based on quintic-polynomial trajectory generation is utilized, since the Computed-Torque tracking method (utilized in [3]) could not be implemented on the robotic system in the CIMLab.

3.4.1 Robot-Motion Initialization

The robot’s motion is initialized as was described in Section 2.4. The motion of the end-effector is defined herein by a displacement vector given by \( \mathbf{d} \), where the direction of \( \mathbf{d} \) is given by Equation (2.13). The constant acceleration model, presented below in Section 3.4.2, is utilized to compute the magnitude of \( \mathbf{d} \), and hence obtain the next end-effector position. The maximum permissible acceleration is utilized.

Robot-motion initialization is terminated when the end-effector reaches a pre-defined percentage of its maximum velocity, or when a pre-defined time threshold has passed.
3.4.2 End-Effector Acceleration Command

In IPNG, the calculation of the acceleration command is carried out using Equation (2.5). Herein, the implementation problem at hand is to determine the end-effector's position as a result of applying the computed IPNG acceleration command for one motion interval, $\Delta t$. Due to the absence of the S-100's jerk-limit specifications and since $\Delta t$ is small, a constant-acceleration model was used in this thesis, at the expense of end-effector acceleration matching between consecutive motion intervals. The model is characterized by a quadratic trajectory:

$$X_R(t) = (X_R)_k + (\dot{X}_R)_k(t-t_k) + \frac{a_{IPNG}}{2}(t-t_k)^2, \quad t \in [t_k, t_{k+1}]$$  \hspace{1cm} (3.7)

where $X_R$ is the positional vector of the robot's end-effector in the Cartesian world coordinate system. Equation (3.7) is utilized to compute the end-effector setpoint at $t_{k+1}$ by setting $t=t_{k+1}$. The robot velocity at $t_{k+1}$ is, thus,

$$(\dot{X}_R)_{k+1} = (\dot{X}_R)_k + a_{IPNG}\Delta t.$$ \hspace{1cm} (3.8)

As discussed in Chapter 2, the IPNG acceleration command must be modified to ensure best utilization of the manipulator's capabilities, without exceeding its specified limits. In Section 2.4, it was shown that, if the robot's dynamic model is utilized, the joint-torque limits must be satisfied when limiting or upgrading $a_{IPNG}$. However, since the manipulator's kinematic constraints are specified herein by maximum end-effector velocity and acceleration, $V_{lim}$ and $A_{lim}$, respectively, these limits have to be satisfied simultaneously.

(i) Upgrading the $a_{IPNG}$

The IPNG acceleration command is upgraded, if its magnitude and the magnitude of the robot velocity, $(\dot{X}_R)_{k+1}$, are less than specified percentages of their maximum values. The acceleration is upgraded in this thesis by adding a component in the direction of LOS, $e_z$:

$$a_c = a_{IPNG} + \beta e_z,$$ \hspace{1cm} (3.9)

where
\[ e_r = \frac{X_T - X_R}{|X_T - X_R|}. \] (3.10)

The problem can be formulated as:

*Given* \( a_{\text{IPNG}}, (\dot{X}_R)_k, e_r, A_{\text{lim}} \) and \( V_{\text{lim}} \), and user-defined percentages \( \alpha_A \) and \( \alpha_V \), find the value of \( \beta \) such that:

\[ \beta = \min (\beta_A, \beta_V), \] (3.11)

where \( \beta_A \) and \( \beta_V \) are computed such that:

\[ |\alpha_c| \leq \alpha_A A_{\text{lim}}, \quad \text{and} \] (3.12)
\[ |(\dot{X}_R)_{k+1}| \leq \alpha_V V_{\text{lim}}, \] (3.13)

respectively. The value of \( \beta_A \) can be obtained by substituting Equation (3.9) into Equation (3.12), Figure 3.2, i.e.,

\[ |a_{\text{IPNG}} + \beta_A e_r| \leq \alpha_A A_{\text{lim}}. \] (3.14)

The value of \( \beta_V \) is obtained by substituting Equations (3.8) and (3.9) in Equation (3.13):

\[ |(\dot{X}_R)_k + (a_{\text{IPNG}} + \beta_V e_r)\Delta t| \leq \alpha_V V_{\text{lim}}. \] (3.15)

Unique solutions for \( \beta_A \) and \( \beta_V \) can be obtained in closed form by noting that the acceleration command should be augmented in the direction of \( e_r \), i.e., \( \beta > 0 \).

Figure 3.2: Upgrading \( a_{\text{IPNG}} \) by adding a component in LOS direction (planar case).
(ii) **Limiting the \( a_{\text{IPNG}} \)**

The robot acceleration must be limited, if either the acceleration or velocity limits are exceeded during the execution of \( a_{\text{IPNG}} \). A sequential process is proposed in this thesis, in which \( a_{\text{IPNG}} \) is first limited if the acceleration limit is violated, and subsequently further limited if the resultant velocity still exceeds the velocity limit.

If the acceleration limit is violated, the \( a_{\text{IPNG}} \) is limited by scaling it down whilst maintaining its original direction:

\[
a_c = K a_{\text{IPNG}} ,
\]

where

\[
K = \frac{A_{\text{lim}}}{|a_{\text{IPNG}}|} .
\]

A different approach is necessary, if the robot velocity at \( t_k+1 \) is expected to exceed the limit, \( V_{\text{lim}} \), since using Equation (3.16) could cause the acceleration command to be scaled down to zero. Figure 3.3 illustrates the above problem schematically for a planar case. It is, thus, proposed to compute an equivalent acceleration command, \( a_{\text{eqv}} \), such that the direction of the robot velocity at \( t_k+1 \) is maintained but its magnitude is reduced to the limit. From the velocity diagram on Figure 3.3:

\[
\dot{x}_r
\]

\[
\dot{x}_n
\]

\[
|\dot{x}| = V_{\text{lim}}
\]

\[
(\dot{x}_k)_{t_k+1} + s_{n\Delta t}
\]

\[
(\dot{x}_c)_{t_k+1} + s_{c\Delta t}
\]

\[
(\dot{x}_c)_{t_k+1}
\]

\[
(\dot{x}_n)_{t_k+1}
\]

\[
(\dot{x}_n)_{t_k+1}
\]

\[
\text{detail}
\]

**Figure 3.3:** Modifying the \( a_{\text{IPNG}} \) due to violation of the velocity limit.
\[
(\dot{X}_R)_{k+1} = \frac{V_{\text{lim}}}{|(\dot{X}_R)_k + a_{\text{IPNG}} \Delta t|}((\dot{X}_R)_k + a_{\text{IPNG}} \Delta t), \quad \text{and} \quad (3.18)
\]
\[
a_{\text{eqv}} = \frac{(\ddot{X}_R)_k - (\dot{X}_R)_k}{\Delta t}. \quad (3.19)
\]

The effective utilization of \(a_{\text{eqv}}\), rather than \(a_{\text{IPNG}}\), merely ensures that the magnitude of the final velocity lies within limits. The modified acceleration command, \(a_c\), is then set as \(a_{\text{eqv}}\).

The overall algorithm for modifying the \(a_{\text{IPNG}}\) is depicted in Figure 3.4. The computation time of the proposed algorithm, for computing \(a_{\text{IPNG}}\) and modifying it if necessary, is \(\sim 1\) ms on a Pentium CPU platform.

3.4.3 Target Tracking

The task of tracking a moving object, in post-IPNG control, is to eliminate the error in position and velocity between the manipulator's end-effector and the object. An example of
such a tracker is the Computed-Torque method used in [3], which is essentially a PD-type tracking method (requiring accurate knowledge of the manipulator’s dynamics). Another common approach is to generate polynomial-trajectories from the current robot state to the target state at a predicted interception point in the future. This is generally the basis of prediction-based interception schema, (e.g., [11], [8]). In these schema, the setpoint which is input to the controller is obtained from the planned trajectory.

Point-to-point motion-planning via trajectory generation has also been utilized, e.g., [17]. In this scheme, a trajectory is generated based on the current state of the robot and the target state predicted one-step ahead. Feddema et al. have proposed a feature-based trajectory generator to track a carburetor gasket in image-based servoing, [19]. In [47], Houshangi uses an auto-regressive prediction model to estimate the target’s position one control sampling period ahead and sets this as the subgoal position for the end-effector. The inputs to the trajectory planner are the current position of the gripper and the target’s position one step ahead. A task-space, trapezoidal-velocity-profile trajectory planner was utilized therein to generate on-line the end-effector position one control sampling period ahead. The desired end-effector position is, thus, computed and fed to the robot’s controller. The quintic-polynomial point-to-point tracker proposed in this thesis for the IPNG hybrid interception scheme bears similarity to this tracking approach.

(i) Quintic Polynomials

Task-space quintic polynomials have been used in trajectory planning for real-time applications (e.g., [11], [31]). Quintic polynomials enable smooth transition from one trajectory spline to another, while satisfying velocity and acceleration constraints, [10]. Cubic polynomials are also commonly used for task-space trajectory generation, e.g., [8], [58]. A comparison of different trajectory generators can be found in [10].

A quintic polynomial is inherently one-dimensional, described as:

\[ X(t) = c_0 + c_1 t + c_2 t^2 + c_3 t^3 + c_4 t^4 + c_5 t^5. \]  

(3.20)

The velocity and acceleration at any time \( t \) can be computed by differentiating Equation (3.20):
The seven independent variables in Equation (3.20), (i.e., the six coefficients \(c_0, \ldots, c_6\) and \(t\)), are uniquely determined, if the initial and final states of the trajectory, as well as the motion time, are specified, [10].

(ii) Target Tracking via Quintic Polynomials

Quintic polynomials have been used extensively for robot-motion planning in our laboratory ([31]), and so they were used in this thesis for target tracking as well. This is achieved by planning a quintic trajectory to generate the path from the current state of the robot to the predicted state of the target one motion time ahead.

Since the quintic polynomial is one-dimensional, three quintic polynomials are necessary to plan the robot's trajectory in 3-dimensional space: one for each of the three orthogonal axes. The computed trajectory is, therefore, expressed as:

\[
X(t) = \begin{bmatrix}
q_x(t) \\
q_y(t) \\
q_z(t)
\end{bmatrix} = \begin{bmatrix}
c_{0x} & c_{1x}t & c_{2x}t^2 & c_{3x}t^3 & c_{4x}t^4 & c_{5x}t^5 \\
c_{0y} & c_{1y}t & c_{2y}t^2 & c_{3y}t^3 & c_{4y}t^4 & c_{5y}t^5 \\
c_{0z} & c_{1z}t & c_{2z}t^2 & c_{3z}t^3 & c_{4z}t^4 & c_{5z}t^5
\end{bmatrix}.
\]

The magnitude of the current robot velocity or acceleration at any time instant can be computed using \(|\dot{X}(t)|\) and \(|\ddot{X}(t)|\), respectively.

In order to determine the coefficients of the quintic-polynomial trajectory, the motion time is required. For time-optimal quintic-motion trajectory, the motion time must be minimized subject to the robot's performance constraints, described herein by the velocity and acceleration limits. Minimum motion time is, thus, achieved when either the velocity or the acceleration limits are reached at some point in the planned trajectory. The problem can, thus, be formulated as, [31]:

*Given the current state of the robot and the predicted state of the target one motion time ahead, and given the velocity and acceleration limits of the robot, \(V_{lim}\) and \(A_{lim}\), find the minimum quintic-polynomial trajectory time, \(t_{qp}\), such that*
\[ t_{qp} = \max (t_{vel}, t_{acc}), \]  
(3.24)

where

\[ t_{vel} = f_v(\text{V}_{\text{lim}}), \quad \text{and} \quad t_{acc} = f_a(\text{A}_{\text{lim}}). \]  
(3.25)

In Equation (3.25), \( f_v(\cdot) \) and \( f_a(\cdot) \) are functions which return such a motion time that either the velocity or the acceleration limit is reached during the execution of the trajectory.

A numerical solution method is required to solve for \( t_{vel} \) and \( t_{acc} \), since the robot and the target are continually in motion, [31]. The algorithm which accomplishes this task was implemented by using a standard iterative numerical search procedure.

The position of the end-effector one motion time ahead is subsequently obtained from the generated trajectory and fed to the robot controller. A new quintic-polynomial trajectory is planned, when new target-motion data becomes available from the vision system. The process is repeated until interception.

It should be remembered that, in the proposed hybrid interception scheme, target tracking is employed only in the last phase of motion in order to decelerate the robot, and so, only very short-term quintic-polynomial trajectories are utilized. The algorithm to determine \( t_{qp} \), and hence generate the trajectory as defined in Equation (3.23), takes 1 - 2 ms on a Pentium CPU computer.

### 3.4.4 Switching from Navigation-Guidance to Tracking

As described in Chapter 2, for time-optimal interception, an on-line selection of the optimal instant at which motion control should be switched from IPNG to the tracking method is necessary. Naturally, the technique to determine the "time-optimal switching point" is dependent on the actual tracking method employed – in this thesis, the Quintic-Polynomial Tracker (QPT).

A pre-defined distance threshold can be utilized for on-line switching. This strategy was adopted by Lin et al. [21] to switch from a coarse-tuning to a fine-tuning interception scheme. This approach does not yield minimum interception time, since the optimal switching point depends on the current robot- and target-motion characteristics. Alternatively, a predictive approach can be used when using the QPT. This would require long-term target-
path prediction to enable the construction and use of a Robot-Travel / Target-Arrival Time diagram similar to the one used in PPE Systems (Figure 1.1). In this thesis, it was decided not to take this approach in the implementation of the IPNG-based interception scheme, since IPNG is targeted for fast-maneuvering targets whose long-term trajectory cannot be reliably predicted without a priori information.

A (near-optimal) solution was adopted in which the current QPT motion time, $t_{qp}$, (computed using Equation (3.24)) is used as the criterion for switching, Figure 3.5.

![Diagram](image)

**Figure 3.5:** Selection of the switching instant.

This approach is based on the assumption that IPNG is superior to the tracking method in bringing the robot towards the target, and thus, tracking is utilized only in the final portion of the intercept period to decelerate the robot to the current state of the target. As the robot closes the distance to the target, the value of $t_{qp}$ will decrease. However, there will be a sudden increase in $t_{qp}$, if the planned quintic trajectory overshoots the target trajectory (e.g., at $t_3$ in Figure 3.5). The minimum point of the $t_{qp}$ curve is, thus, utilized in this thesis to determine the switching point in an on-line manner.

In implementing the above switching strategy, the values of $t_{qp}$ computed on-line are smoothed to reduce the effect of noise. Various conventional smoothing techniques were considered, namely moving averages, recursive exponential smoothing and trend-adjusted
exponential smoothing, [59]. Recursive exponential smoothing was adopted, due to the simplicity of computation and the possibility of sensitivity adjustment (to random variations) by the selection of the smoothing constant, $\alpha$. Recursive exponential smoothing is computed using:

$$\left(\tilde{t}_{qp}\right)_k = \alpha_q (t_{qp})_k + (1 - \alpha_q) \left(\tilde{t}_{qp}\right)_{k-1},$$  \hspace{1cm} (3.26)

where the tilde denotes smoothed data. Switching is performed when:

$\text{Condition (1): } (\tilde{t}_{qp})_k > (t_{qp})_{k-1}, \text{ or }$  \hspace{1cm} (3.27a)

$\text{Condition (2): } (t_{qp})_k > (t_{qp})_{k-1} \text{ for two consecutive steps. }$  \hspace{1cm} (3.27b)

Condition (2) prevails, if an unsuitable selection of $\alpha_q$ is made.

The switching algorithm typically requires 2 ms to compute a Pentium CPU computer.

### 3.4.5 Overall Robot-Motion Planning Algorithm

The primary steps required to compute the robot motion command for the interval $[t_k, t_{k+1}]$ are:

* **Step (1)** The $a_{IPNG}$ is determined based on the state of the robot at time $t_k$ (i.e., at the end of the current motion interval) and the target's predicted state at $t_k$;

* **Step (2)** The final state of the robot at $t_{k+1}$ is computed (by modifying the $a_{IPNG}$ as necessary);

* **Step (3)** The $t_{qp}$ for the interval $[t_{k+1}, t_{k+2}]$ is determined based on the robot state at $t_{k+1}$ and the target's predicted state at $t_{k+2}$;

* **Step (4)** The conditions for switching, as defined in Equations (3.27a) and (3.27b), are checked:

  If either of the conditions is satisfied,

  * **Step (4a)** Control is switched to tracking and the robot state for the motion interval $[t_k, t_{k+1}]$ is re-computed using the QPT technique, (based on the robot state at time $t_k$ and the predicted target state at time $t_{k+1}$);

  Otherwise,

  * **Step (4b)** The final robot state computed in Step (2) is applied to the robot.
3.5 Interception-Planning Simulations

The following examples were generated using the algorithms presented in this chapter. Target-motion data was obtained via simulations. The tolerances of interception were set as: \( tol_p = 10 \text{ mm} \) and \( tol_v = 10 \text{ mm/s} \). The navigation gain was set at 3, and \( k \) in Equation (2.13) was set at 3. \( \alpha_V \) and \( \alpha_A \) were set at 0.9 and 0.8 respectively. The smoothing constant, \( \alpha_s \), was set at 0.45. The motion interval duration was set at 220 ms, which corresponds to the fastest rate which the Karel controller allows for communication of trajectory data.

In the first example, the target is moving on a sinusoidal path defined as:

\[
X_T = \begin{bmatrix}
400.0 + 30.0 t \\
150 - 200.0 \sin \left( \frac{2.5\pi}{10.0} t \right) \\
0.0
\end{bmatrix} \text{ mm.}
\]  

(3.28)

The end-effector is initially at rest at the point \([0.0, 0.0, 150.0]^T \text{ mm} \). The robot’s permissible velocity and acceleration limits were set as 300 \( \text{ mm/s} \) and 400 \( \text{ mm/s}^2 \) respectively. A planar view of the intercept is illustrated in Figure 3.6, whereas Figure 3.7 depicts the variations of the \( x \), \( y \) and \( z \) components of the position and velocity of the target and the end-effector versus time. The hybrid IPNG-QPT technique yields an interception time of 7.3 s, whereas the pure QPT technique would yield an interception time of 10.3 s.

![Figure 3.6: Interception of a target moving on a sinusoidal trajectory (planar view).](image-url)
Figure 3.7: Position and velocity variation versus time for end-effector and target.

Figure 3.8 shows the variation of the interception time versus the switching time. One should remember, however, that during run-time, the value of the interception time is not available, and the selection of the switching time is consequently based on the values of $t_{qp}$. The interception time curve is given herein to provide a basis of comparison of the effect of the switching time. Figure 3.8 also depicts the variation of the $t_{qp}$ versus the switching time. It can be seen that, $t_{qp}$ decreases continually as the robot approaches the target, however, there is a sharp increase to the right of the minimum value of $t_{qp}$, which occurs at 5.7 s. One can note
that, in real-time, the worst-case interception time obtained using the hybrid IPNG-QPT scheme would be equal to that of the pure QPT method if switching occurs immediately following motion initialization with no IPNG phase.

![Graph showing variation of \( t_{\text{int}} \) and \( t_{\text{qp}} \) versus switching time](image.png)

**Figure 3.8:** Variation of \( t_{\text{int}} \) and \( t_{\text{qp}} \) versus switching time for the example shown in Figure 3.6.

Figure 3.9 illustrates the planar interception of a target moving on a linear trajectory. The robot is initially at rest at \([0.0, 0.0, 0.0]^T\) mm. The target’s trajectory is defined as:

\[
X_T = \begin{bmatrix} 700.0 - 30.0 t \\ 0.0 - 55.0 t \\ 0.0 \end{bmatrix} \text{ mm.} \quad (3.29)
\]

The robot’s permissible velocity and acceleration limits were set as 100 mm/s and 150 mm/s\(^2\) respectively. The proposed hybrid method yields an interception time of 9.7 s, whereas the pure QPT method yields an interception time of 20.2 s. The variations of the \(x\) and \(y\) components of the position and velocity of the robot and the target versus time are depicted in Figure 3.10. Figure 3.11 shows the variation of the interception time and \( t_{\text{qp}} \) versus the switching time. This curve shows a rapid initial decrease of \( t_{\text{int}} \); however, towards the end of the IPNG phase of interception, switching at different instances yield practically the same interception time (within a tolerance range of one \( \Delta t \)). As expected, the \( t_{\text{int}} \) curve subsequently increases rapidly. This would occur because the robot’s trajectory would overshoot the target’s
trajectory. Using the proposed algorithm, switching occurs at 5.7 s, at the minimum value of $t_{qp}$.

Figure 3.9: Interception of a target moving on a linear trajectory.

Figure 3.10: Position and velocity variation versus time for end-effector and target.
Figure 3.11: Variation of \( t_{\text{int}} \) and \( t_{\text{qp}} \) versus switching time for the example depicted in Figure 3.9.

As has been described in Chapter 2, the \( a_{\text{IPNG}} \) tries to turn the relative velocity between the robot and the target onto the LOS. While this is achieved, the robot moves with equal velocity perpendicular to the LOS whilst it closes the relative distance. Thus, the \( a_{\text{IPNG}} \) tries to turn the robot toward an interception point ahead of the target’s current position. However, when the QPT method takes over control, it tries to turn the robot to the position of the target one-step ahead, rather than trying to maintain a constant heading with the target (while closing the relative distance). (This accounts for the change in direction of end-effector motion which occurs after switching in Figure 3.9). Thus, the robot would subsequently have to catch up with the target as the latter moves along its trajectory. This causes an increase in the interception time.

The above mentioned shortcomings in the QPT were examined for improvement. A new tracking algorithm which potentially may improve upon the QPT is discussed in Chapter 5.
3.6 Summary

In this chapter, the robot-motion planning algorithms which were utilized in the implementation of the IPNG-based hybrid interception system were presented. In these algorithms, the robot’s dynamics are bypassed (due to unavailability), where the robot’s capabilities are characterized by its end-effector’s velocity and acceleration limits. The IPNG technique is implemented as discussed in Chapter 2; however, the modification of the acceleration command needed to be carried out herein subject to the velocity and acceleration limits, rather than torque limits. The CT tracking technique was also changed to a Quintic-Polynomial Tracking (QPT) technique. Thus, a new switching algorithm had to be proposed. It yields near-optimal interception time. The simulation results show that the proposed hybrid interception scheme yields faster interception than the QPT method alone.

Chapter 4 will present the architecture of the actual experimental interception system developed in the CIMLab.

4.1 Introduction

In Chapter 3, the motion-planning algorithms developed in this thesis for the hybrid Ideal Proportional Navigation Guidance - Quintic-polynomial Tracking (IPNG-QPT) interception scheme were discussed. This chapter focuses on the experimental system developed for the real-time testing of these algorithms in the CIMLab. The chapter is organized into two main sections: in Section 4.2, the experimental system configuration is presented, namely, the hardware and function of each sub-system, as well as a brief discussion of implementation issues pertaining to the integration of these sub-systems. Section 4.3 presents the experimental results obtained when intercepting both fast-maneuvering and slow-maneuvering targets.

4.2 System Configuration

The experimental IPNG-QPT moving-object interceptor test-bed comprises three primary sub-systems that utilize four main modules, Figure 4.1: the vision module, which acquires target-images in real-time; the prediction module, which performs target-motion prediction; the motion-planning module, which plans the robot trajectory on-line; and, the execution module.

Within the Object-Tracking Sub-system, target tracking is achieved using a (Hitachi 30 Hz) CCD camera (equipped with a Canon 25 mm lens) and a (Matrox 640B) frame grabber, which acquire 640x480 pixel digital images with 256 grey-level resolution. The host is an Intel 486 DX2 66 MHz PC. The Interception-Planning Sub-system is a Pentium 200 MHz PC. This sub-system hosts both the target-motion prediction and the robot-motion planning modules. This enables the predictor to efficiently provide the planner with estimated target motion at the
latter's request. The Execution Sub-system consists of the GMFanuc S-100 robotic manipulator and its Karel controller. Data transfer between the different sub-systems is achieved using serial communication at 9600 baud.

Object motion is achieved using the xy-table of a NC milling machine.

![Diagram](image)

**Figure 4.1:** Architecture of the experimental IPNG-QPT interception system.

### 4.2.1 Object-Tracking Sub-System

The CCD camera is set up in a fixed-camera configuration. It is mounted at 1.75m above the plane of motion of the target, at an angle of approximately 8° with the normal to the plane. The current configuration yields a field of view of approximately 47x58 cm² at the target's motion plane. This corresponds to a resolution of approximately 0.98 mm/pixel along the x direction of target motion, and 0.9 mm/pixel along the y direction of motion.
As is usually the case in most moving-object-interception systems reported in the literature (e.g., [27], [10]), a structured environment is utilized herein to simplify the image-processing computational burden. The target consists of a white circular marker moving on a darker background, permitting target localization in the scene by using grey-level thresholding and centroid computation. The threshold value utilized to convert the gray-scale images into binary is a priori determined experimentally. The windowing approach (e.g., [11]) was adopted in this thesis, whereby only a small rectangular window located at the current marker location is processed to save time. The window is multiplied in size if the target is not located within the window. Once the centroid of the object has been located in the image plane, its pixel coordinates are transformed to the world coordinate system of the robot. This is achieved on-line using a priori knowledge of the camera-calibration model.

The CCD camera was calibrated off-line using the non-coplanar calibration technique proposed by Tsai [56]. This technique enables the computation of both the camera’s intrinsic and extrinsic parameters. The intrinsic parameters include the focal length, the lens distortion coefficient, the uncertainty scale factor for the horizontal axis scanning (due to camera scanning and acquisition timing errors), and the 2D camera-image center coordinates. The six translation and rotation parameters defining the 3D rigid body transformation from the world coordinate system to the camera coordinate system constitute the extrinsic parameters.

Using the technique proposed in [56], the model parameters are determined using a set of points whose coordinates with respect to the world coordinate system are known and whose image coordinates can be measured. This was achieved using an end-effector mounted plate with precisely machined holes. The optimization process that yielded the estimation of the calibration parameters for the Hitachi camera in the interception test-bed was achieved using the calibration C-code software package developed by Willson [60]. The same software was used to gauge the accuracy of the model parameters. The calibration model yields maximum errors of 1 mm and 2 mm along the X and Y directions of motion of the target, respectively, [61]. These calibration errors are equivalent to approximately 0.2% and 0.35% of the field of view, and are quite acceptable for our experiments.
The algorithms of the vision module were written in C on a Borland C/C++ compiler. On the current sub-system, the process of acquiring an image and processing it to generate the world coordinates of the target centroid utilizes approximately 66 ms.

4.2.2 Interception-Planning Sub-System

The purpose of this sub-system is to utilize the target positional data supplied by the Object-Tracking Sub-system to generate a position command for the Execution Sub-system. The Object-Tracking Sub-system provides fresh target data every 66 ms, whereas the Execution Sub-system requires positional setpoints approximately every 220 ms. Thus, whenever fresh target data becomes available, the target’s trajectory estimates are updated by the KF and the end-effector setpoints are re-generated using the most recent target trajectory data.

The process of updating the KF target-motion model, performing one step-ahead predictions (as required by the planner) and generating the end-effector setpoint (using the motion-planning algorithms) requires, typically, 2-3 ms on the interception-planning sub-system. The algorithms were written in C on a Borland C/C++ compiler.

4.2.3 Execution Sub-System

The Execution Sub-system comprises the GMFanuc S-100 robot and its Karel controller. The Karel controller is instructed to perform joint-interpolated, continuous motion between successive end-effector setpoints. Trajectory setpoints are provided by the Interception-planning Sub-system at the Karel controller’s request. Data is transmitted in binary format, as opposed to ASCII format, since the former yields faster communication. Serial communication at 9600 baud was utilized; the fastest rate which the controller permits. The motion interval is set at 220 ms – the robot does not manage to perform the desired motion if shorter intervals are used. The Karel high-level language is used to encode the robot-motion algorithms on the Karel controller.

To ensure that the robot is tracking the desired setpoints, a Karel high-level instruction, curpos, is used to return the current end-effector position. This allows the comparison of the
actual end-effector path to the desired setpoints determined by the planner: for the acceleration and speed limits utilized in the experiments, the errors were very small (within 1 mm per point).

4.2.4 Object Motion

In the moving-object-interception test-bed in the CIMLab, object motion is achieved using programmed motions of the numerically-controlled xy-table of a Bridgeport Series I milling machine. The NC machine is equipped with a Bandit digital controller. It provides precise target trajectories with rapid accelerations and decelerations, and a maximum speed of approximately 35 mm/s.

4.2.5 System Integration

The target-tracking, motion-planning and motion-execution processes run asynchronously on their respective sub-systems. Serial communication is used to link the sub-systems. When new target data is available, the vision module informs the motion-planning module of this availability and communicates this data at the planner's request. Similarly, data is sent to the robot controller at its request. When either of these events has occurred (i.e., new target data has been received, or robot setpoint has been sent), the motion-planning algorithm is employed to generate a new end-effector setpoint based on the most recent target data.

A system timer resides within the Interception-Planning Sub-system, since this sub-system monitors the events within both the Object-Tracking and the Execution Sub-systems. The compiler provides a timer with a coarse resolution of 18 Hz (i.e., the time is updated every 55 ms). However, since a faster timing resolution was required in this thesis, it was achieved by speeding up the timer via software to a resolution of 1 ms, [62].

A flowchart that illustrates the execution of events within all sub-systems is depicted in Figure 4.2.
**Figure 4.2:** Flowchart for the hybrid IPNG-QPT interception system.
4.2.5.1 Communication Modules

Communication links are needed between the Object-Tracking Sub-system and the Interception-Planning Sub-system, and between the Interception-Planning and the Execution Sub-systems. Data is transferred serially through RS-232 ports at 9600 baud. A communication module was, thus, incorporated into each sub-system. The communication modules on the Object-Tracking and the Interception-Planning Sub-systems were written in C, making use of the serial communication routines provided by the Borland C/C++ compiler. On the Execution Sub-system, the serial communication handler provided by the Karel operating system was utilized.

4.2.5.2 System Initiation

The experimental system is started by initiating the hardware and executing the software programs. The communication links between the sub-systems are established by resetting the communication ports and exchanging the initial handshaking signals. On the Object Tracking Sub-system, the first image is initially thresholded and scanned completely by reading pixels at equally spaced intervals to locate the marker. Once the marker is located and its centroid determined, the search window is initially centered at these centroid coordinates.

Target motion is manually started by executing the programmed motion of the xy-table on the milling machine. When the Object Tracking Sub-system detects target motion (i.e., centroid displacement greater than a predefined threshold), it informs the planner to trigger the system timer. On the Interception-Planning Sub-system, the first images are utilized to initialize the KF, [31]. Following KF initialization, the motion-planning algorithm for the hybrid IPNG-based interception scheme takes over control and starts sending trajectory data to the robot. This marks the start of the interception timer.

4.2.5.3 System Delays

The planner must take into account all system delays when generating robot commands; otherwise, the robot would arrive at the interception point later than the target, and hence miss the target. The most significant system delay is introduced by the Karel controller. For
continuous motion from a given point to a second point, the Karel controller requires knowledge of the third point (to enable it to plan the transition at the second trajectory point). Thus, whenever the Execution Sub-system requests data, the setpoint of the end-effector two motion intervals ahead must be sent by the Interception-Planning Sub-system. This constitutes an initiation delay of approximately 440 ms at the start of interception. Another significant system delay is due to the vision module, which sends target motion data acquired 66 milliseconds before. The prediction module on the Interception Planning Sub-system was utilized to compensate for these system delays by predicting the target state ahead.

4.3 Experimental Results

The experimental set-up discussed above was employed to verify the hybrid IPNG-QPT method to intercept moving targets. For simulation purposes, the robot’s speed limit was set at a much smaller value (maximum of 70 to 120 mm/s) than its actual permissible limit (maximum of 1000 mm/s), due to the velocity limitation of the xy-table of the milling machine (maximum of 35 mm/s), on which the target is displaced.

4.3.1 Slow-Maneuvering Objects

In the following two experiments, the effect of the switching point on the interception time is shown. The tolerances of interception, \( tol_p \) and \( tol_o \), were set at 10 mm and 10 mm/s respectively, and the end-effector’s velocity and acceleration limits were set at 70 mm/s and 150 mm/s².

Figure 4.3 shows a planar view of the interception of the target moving on a linear trajectory with a cruising speed of approximately 30 mm/s. The end-effector is initially at rest at \([200 \ -300 \ 50]^T\) mm. The \(x\), \(y\) and \(z\) components of the position of the robot’s end-effector under IPNG-QPT control (as sent by the planner to the robot controller) and of the target (as received from the Object-Tracking Sub-system) are shown in Figure 4.4. Figure 4.4 also shows the velocity components of the target (estimated via the KF) and of the end-effector.
Figure 4.3: Interception of a target moving on a linear trajectory (planar view).

Figure 4.5 shows the variation of the interception time versus the switching time. In our experiments, the switching time was manually varied at 0.5 s intervals. For the above target motion, one notes an initial rapid decrease in interception time, and subsequently only a small decrease (within one motion interval) towards the end of the IPNG phase of motion. As expected, the curve of $t_{qp}$ decreases continually as the end-effector gets closer to the target. Using the method for automatic switching from IPNG to QPT, switching occurs at 8 s (at the minimum of the $t_{qp}$ versus switching time curve), yielding an optimal interception time of 9.6 s.
Figure 4.4: Position and velocity variation versus time for the end-effector and the target.

Figure 4.5: Variation of $t_{in}$ and $t_{eq}$ versus switching time for the example depicted in Figure 4.3.
Figures 4.6 and 4.7 show the interception of the target when moving on a circular trajectory. Figure 4.8 shows the variation of the interception time versus the switching time. Using the proposed switching algorithm, motion control is switched to tracking at 5.8 s, at the minimum value of the $t_{qp}$ versus switching time curve. This yields an interception time of 7.6 s. However, in this case, the minimum of the $t_{int}$ curve occurs at a switching time of 6 s, i.e., one motion interval after the switching time at which the minimum $t_{qp}$ occurs. Switching at 6 s would yield an interception time of 7.2 s. Thus, one may state that the proposed switching algorithm did not yield the optimal interception time. Still, however, the resulting interception time represents a decrease of more than 2.5 s over the interception time which the pure QPT method would yield.

![Figure 4.6: Interception of a target moving on a circular trajectory (planar view).](image-url)
Figure 4.7: Position and velocity variation versus time for the end-effector and the target.

Figure 4.8: Variation of $t_{int}$ and $t_{qp}$ versus switching time for the example depicted in Figure 4.6.
4.3.2 Fast-Maneuvering Objects

In the following examples, the interception of a fast-maneuvering target is considered. Figure 4.9 shows a planar view of the interception of a target which is initially moving along a circular trajectory. It then stops and rapidly reverses its direction, moving along a circular path in the opposite direction. In this example, the robot's velocity and acceleration limits were set at 70 mm/s and 120 mm/s² respectively. Figure 4.10 shows the x, y and z components of the position and velocity of the robot's end-effector under IPNG-QPT control and of the target.

The algorithm for automatic switching succeeds in detecting the minimum value of $t_{qp}$, which occurs at a switching time of 7.6 s. Figure 4.11 shows that switching after 7.6 s would yield a sharp increase in the interception time. This would occur because the robot overshoots the target's trajectory. In this case, however, the minimum interception time does not occur just prior to the sharp increase of interception time curve; it occurs at a switching time of 5.5 s. This is due to the discontinuity and sudden reversal in the target's trajectory, which occurs at approximately 6.7 s.

![Figure 4.9: Interception of a target moving on a circular trajectory and changing its direction of motion (planar view).](image)
Switching at 5.5 s would yield interception at 8.5 s, whereas the automatic switching algorithm yields an interception time of 9.1 s (at a switching time of 7.6 s). The interception time obtained by the automatic switching, though not optimal, still represents a significant decrease from the interception time which the pure QPT method would yield, (i.e., 12.5 s).
Figures 4.11 and 4.13 depict the interception of a target travelling along a stop-and-go trajectory: the target is initially moving along a circular path, it decelerates rapidly to rest, and after 1.5 s accelerates rapidly on a linear path. For this example, the robot’s velocity and acceleration limits were set at 120 mm/s and 200 mm/s². Using the PNG-QPT method, interception occurs at 7.0 s, whereas the pure QPT method yields an interception time of 9.1 s.

Figure 4.11: Variation of $t_{\text{int}}$ and $t_{\text{ap}}$ versus switching time for the example depicted in Figure 4.9.

Figure 4.12: Interception of a target travelling along a stop-and-go trajectory (planar view).
4.4 Summary

In this chapter, the architecture of the experimental test-bed developed within this thesis to verify the IPNG-based interception scheme has been presented. Implementation issues pertaining to each sub-system in the test-bed have been discussed. The experimental results illustrate the capability of the hybrid IPNG-QPT scheme to intercept targets faster than would the pure QPT method.
Chapter 5: Discussion

5.1 Introduction

In the previous chapters, the implementation of the hybrid IPNG-based interception scheme was presented. It was shown that the proposed navigation technique brings the end-effector towards the target as fast as possible. However, it was noted that the tracking technique must take over control of the end-effector’s motion so that the target velocity matching is also ensured. In this chapter, the (QPT) tracking method is modified such that it succeeds in closing the distance to the target in a similar manner as IPNG, whilst still ensuring target velocity matching at the interception point, i.e., perform navigation-based tracking of the moving target. This method has been developed for preliminary examination purposes, to provide a basis for recommendation for future work in this area.

Before presenting the modified tracking technique, a brief discussion of the effect of bypassing the robot’s dynamics, during the experiments, on the robotic IPNG is given.

5.2 Effect of Unavailability of Robot’s Dynamics on Robotic IPNG

In [3], it was proposed to use the robot’s dynamic model to obtain the joint torques required to move the end-effector with the computed acceleration command. Namely, during the IPNG-phase of interception, the IPNG acceleration command, $a_{IPNG}$, is used to generate the robot’s motion commands. The $a_{IPNG}$ is upgraded or limited subject to the robot’s joint-torque limits. The $a_{IPNG}$ is upgraded by adding an acceleration component in the direction of the LOS. The magnitude of this acceleration component is computed by mapping the joint torques to the end-effector acceleration at the current robot configuration.

If the $a_{IPNG}$ is modified subject to a fixed acceleration limit for any state of the manipulator in the operating region, under-utilization of the actuators would result ([63]). The global end-effector acceleration limit represents a maximal lower bound on the magnitude of the acceleration that can be achieved at the end-effector for any state in the operating region, [64].
In this thesis, due to the unavailability of the dynamic model of the CIMLab's robot, the IPNG acceleration command is modified subject to fixed velocity and acceleration limits of the end-effector. Thus, full-utilization of the manipulator's speed capability is not achieved, [39]. This results in longer interception times than those which could be obtained if the manipulator's dynamic constraints were taken into account. However, this does not negate the achievement of the primary objective of this thesis: Illustration of the effectiveness of the IPNG-based interception method originally proposed by Mehrandezh et al. [3].

5.3 Navigation-Based Tracking - Motivation

Navigation-based techniques bring the interceptor close to the target as fast as possible without attempting to match the target's velocity. This feature enables the interceptor to close the distance between the interceptor and the target in a very fast manner. However, these techniques are designed to bring the interceptor in a collision course with the target, rather than to ensure a smooth grasp. Thus, for robotic interception schema, navigation-based techniques must be followed by a tracking method which ensures stable tracking and smooth grasping, [3]. The hybrid IPNG-QPT scheme implemented in this thesis is one such example.

In the hybrid IPNG-QPT interception scheme, the on-line selection of the switching instant, at which control changes from the navigation technique to the tracking method, has a considerable effect on the interception time obtained. An unoptimal selection of the switching point could result in a longer interception time.

It would be very advantageous to develop a tracking technique which can yield fast interception of maneuvering targets in a similar manner to navigation-based techniques, while ensuring target-velocity matching at the interception point. Such a "navigation-based tracker" would not try to match the target's position and velocity when the target is still far, but will automatically decelerate the interceptor to the target's velocity at the interception point in a time-optimal manner. Since the interception scheme will consist of one method, the interception time would also be independent of the selection of a switching instant as is the case in hybrid interception schema.
In the next section, the QPT algorithm described in Chapter 4 is modified to perform target-tracking in a similar fashion to the IPNG technique. When the modified QPT takes control, it continually tries to turn the relative velocity onto the LOS as the distance is closed, while ensuring that velocity matching is achieved at the interception point.

5.4 Modified Quintic-Polynomial Tracking Method

A novel modified Quintic-Polynomial Tracker (mod-QPT) is examined below. The QPT algorithm is modified to perform navigation-based tracking. The aim is to conduct a preliminary investigation to determine the potential of such a tracking algorithm in improving upon the QPT in the experimental, IPNG-based interception system. The scope of this section and the preliminary results obtained is to provide a basis for recommendations for future work in this area.

In this thesis, the mod-QPT is examined for 2D interception only. The objective of the mod-QPT is to bring the end-effector into a pre-grasp position with the target, matching the location and velocity of the target to within pre-defined tolerances, $tol_p$ and $tol_v$. As is the case for the QPT, it is assumed that the robot has a clear speed and acceleration superiority over the target.

The same experimental set-up developed for the IPNG-QPT method was used to obtain preliminary results on the performance of the mod-QPT.

5.4.1 Basic Concept

The IPNG acceleration command tries to turn the relative velocity onto the LOS with utmost effort while closing in on the target, Figure 5.1. In a similar fashion, the mod-QPT tries to turn the relative velocity onto the LOS while tracking the target. However, it must ensure that velocity matching along the LOS direction is also achieved at the interception point, for a smooth grasp. Thus, the mod-QPT:

(1) turns the relative velocity onto the LOS, i.e., achieves zero relative velocity in a direction perpendicular to the LOS as fast as possible, while,
(2) ensuring that, as the end-effector closes the distance to the target, the relative velocity is reduced to zero at the interception point.

![Diagram of relative velocity parallel to the LOS.](image)

**Figure 5.1:** Relative velocity parallel to the LOS.

### 5.4.2 Formulation of the Tracking Problem

Consider an orthogonal frame consisting of the unit vectors $e_r$, $e_\phi$, and $e_z$ located at the origin of the Cartesian world coordinate system $(x, y, z)$, Figure 5.2. This system of coordinates is defined such that $e_r$ is parallel to the LOS at time $t_k$. The coordinate system is assumed to be fixed in space for the motion interval $[t_k, t_{k+1}]$, (i.e., the time derivatives of the unit vectors are zero). Planar motion in the Cartesian $xy$-plane is considered herein, (and $e_z = -z$).

In the original QPT, target tracking is achieved by planning a quintic trajectory to generate the path from the current state of the end-effector to the predicted state of the target one motion time ahead. The next end-effector position is, then, obtained from the generated trajectory.
Figure 5.2: Geometry of the modified Quintic-Polynomial tracking method.

In the modified QPT, target tracking is achieved in the 2D $e_x e_{\phi}$ - plane, by generating two independent trajectories for the end-effector: one along the $e_r$ axis and another along $e_{\phi}$:

(i) **Along $e_r$** - a one-dimensional trajectory is generated from the current state of the end-effector (projected onto the $e_r$ axis) to the predicted target state, (along $e_r$), one motion interval ahead. The motion time of this trajectory for the time interval $[t_k, t_{k+1}]$ is denoted by $t_{\text{modQPr}}$;

(ii) **Along $e_{\phi}$** - the sub-goal is to obtain zero relative velocity. Thus, a trajectory is generated such that the end-effector can match, as fast as possible, the target’s velocity (and acceleration) components projected along $e_{\phi}$, predicted one motion time ahead. In this direction, the end-effector will either, accelerate or decelerate from its current speed to the target’s speed, or, move with constant velocity if it is already moving with the target’s speed. The motion time of this trajectory, for the time interval $[t_k, t_{k+1}]$, is denoted by $t_{\text{modQP}{\phi}}$. The motion times of the two trajectories are decoupled, i.e., $t_{\text{modQP}{\phi}}$ is not necessarily equal to $t_{\text{modQPr}}$. For example, it is expected that when the mod-QPT takes over
control from IPNG, the relative velocity along \(e_\phi\) is already very small (or zero), and so the robot can soon achieve the target's velocity (i.e., \(t_{modQP}\) would be small).

### 5.4.2.1 Mapping Position, Velocity and Acceleration Vectors

Since tracking is performed in the \(e,e_\phi\) - plane, the robot's and target's states are mapped into this frame to compute the trajectories. For the motion interval \([t_k, t_{k+1}]\), the \((e_r, e_\phi, e_z)\) frame is constructed based on the end-effector and target positions at \(t_k\). The target's predicted state one motion time ahead is also mapped to enable target tracking. Subsequently, the two trajectories are generated and the end-effector's state for the end of the motion interval is obtained from the computed trajectories. This state is then mapped back to the Cartesian frame to obtain the next setpoint for the GMFanuc S-100 manipulator.

The rotational operator defining the rotation from the Cartesian \(xy\) - plane into the \(e,e_\phi\) - plane is given as, Figure 5.3:

\[
R = \begin{bmatrix}
\cos \theta_{LOS} & \sin \theta_{LOS} \\
\sin \theta_{LOS} & -\cos \theta_{LOS}
\end{bmatrix}.
\]  

(5.1)

\[\theta_{LOS}\] denotes the angle between the LOS and the \(x\) axis of the Cartesian world coordinate system. Also,
\[
\cos \theta_{\text{LOS}} = \mathbf{x} \cdot \hat{\mathbf{r}} = \hat{r}_x,
\]

and,

\[
\sin \theta_{\text{LOS}} = (\mathbf{x} \times \hat{\mathbf{r}})_z,
\]

where the "\(\cdot\)" and the "\(\times\)" symbols denote the scalar and vector products, respectively, and \(\hat{\mathbf{r}}\) denotes the unit relative position vector.

### 5.4.2.2. Generating the Trajectories Along the \(e_r\) and \(e_{\phi}\) Axes

(i) Generating the trajectory along \(e_r\)

The trajectory for the end-effector displacement along \(e_r\) is generated from the current state of the end-effector to the target's state predicted one-step-ahead, i.e., at \(t_{k+1}\). The initial conditions are the robot's position, velocity and acceleration components projected along the \(e_r\) axis, (i.e., \((X_R)_k\), \((\dot{X}_R)_k\), and \((\ddot{X}_R)_k\)), whereas the final boundary conditions are the target's position, velocity and acceleration components predicted for \(t_{k+1}\), (i.e., \((X_T)_{k+1}\), \((\dot{X}_T)_{k+1}\), and \((\ddot{X}_T)_{k+1}\)).

A one-dimensional quintic polynomial is utilized to generate the end-effector trajectory, Equation 3.20, [10]. As discussed in Section 3.4.3, the motion time is required to fit a quintic polynomial. The motion time, \(t_{\text{modQP},r}\), however, must be minimized subject to the velocity and acceleration constraints. A numerical search algorithm was utilized, herein, to obtain the minimum \(t_{\text{modQP},r}\), [31], given the maximum permissible velocity and acceleration for end-effector motion along the \(e_r\) direction of motion, \(V_{\text{lim},r}\) and \(A_{\text{lim},r}\). Subsequently, the quintic trajectory is generated using \(t_{\text{modQP},r}\). The end-effector state along \(e_r\) for the next motion interval is then obtained from the generated trajectory.

(ii) Generating the trajectory along \(e_{\phi}\)

Along \(e_{\phi}\), the sub-goal is to achieve the target's velocity as fast as possible. Thus, a trajectory for the end-effector displacement along \(e_{\phi}\) is generated such that the end-effector can match the target's velocity (and acceleration) predicted one-step-ahead. The initial conditions are the robot's position, velocity and acceleration components projected along the \(e_{\phi}\) axis (i.e.,
\((x_R)_k, (\dot{x}_R)_k, \text{ and } (\ddot{x}_R)_k\), whereas the final conditions are the target's velocity and acceleration components predicted for \(t_{k+1}\), i.e., \((\dot{x}_T)_k\), and \((\ddot{x}_T)_k\). Since there is one less boundary condition, (i.e., target position at \(t_{k+1}\)), the coefficients of the quintic polynomial cannot be uniquely determined. A quartic polynomial trajectory generator is, thus, utilized:

\[
\dot{x}_R(t) = c_0 + c_1 t + c_2 t^2 + c_3 t^3 + c_4 t^4, \quad t \in [0, t_{modQP\phi}].
\] (5.3)

The velocity and acceleration at any time \(t\) can be computed by differentiating Equation (5.3):

\[
\dot{x}_R(t) = c_1 + 2c_2 t + 3c_3 t^2 + 4c_4 t^3, \quad (5.4a)
\]

\[
\ddot{x}_R(t) = 2c_2 + 6c_3 t + 12c_4 t^2. \quad (5.4b)
\]

The five coefficients, \(\{c_0, \ldots, c_4\}\), are uniquely determined by substituting the initial and final conditions of the trajectory, given the motion time. A numerical search algorithm was utilized to obtain the minimum \(t_{modQP\phi}\), given the maximum permissible velocity and acceleration, \(V_{lim\phi}\) and \(A_{lim\phi}\). The end-effector state is, then, obtained by setting \(t = \Delta t\) in Equations (5.3) and (5.4).

(iii) Dividing the available robot velocity and acceleration

Since the trajectories for the end-effector's motion along the two perpendicular axes are generated independently, the end-effector's velocity and acceleration limits, \(V_{lim}\) and \(A_{lim}\), must be divided for the two trajectories. For time-optimal motion, an optimization process would be required to ensure that the division process will yield minimum \(t_{modQP\phi}\) and \(t_{modQP\phi}\). However, a simplified procedure for dividing the velocity and acceleration limits is proposed herein for the hybrid IPNG-mod-QPT under examination. (It is assumed that the end-effector is in motion under IPNG control when the mod-QPT initially takes over control). The method is described in algorithmic form below.

\textit{Step 1:} Set \(V_{lim\phi} = \max((\dot{x}_R)_k, (\dot{x}_T)_k)\)

and \(A_{lim, x} = \alpha_A A_{lim}, \quad 0 < \alpha_A < 1\) (typically 0.8)
Step 2: Set \( V_{\text{lim} r} = \alpha_V \sqrt{V_{\text{lim} r}^2 - V_{\text{lim} \phi}^2} \), \( 0 < \alpha_V < 1 \) (typically 0.9)

Step 3: If \( V_{\text{lim} r} > \max(\langle \dot{X}_R \rangle_k, \langle \dot{X}_T \rangle_{k+1}) \),

generate a trajectory along \( e_r \) using \( V_{\text{lim} r} \) and \( A_{\text{lim} r} \).

Find the end-effector acceleration along \( e_r \) at \( t_{k+1} \), \( \langle \ddot{X}_R \rangle_{k+1} \), and set

\[
A_{\text{lim} \phi} = \sqrt{A_{\text{lim} r}^2 - \langle \dot{X}_R \rangle_{k+1}^2}.
\]

Otherwise,

Go to Step 4.

Step 4: Set \( V_{\text{lim} r} = \max(\langle \dot{X}_R \rangle_k, \langle \dot{X}_T \rangle_{k+1}) \) and generate a trajectory along \( e_r \) using \( V_{\text{lim} r} \) and \( A_{\text{lim} r} \).

Find the end-effector velocity and acceleration components at \( t_{k+1} \), \( \langle \dot{X}_R \rangle_{k+1} \) and \( \langle \ddot{X}_R \rangle_{k+1} \).

Limit the sub-goal velocity for the end-effector trajectory along the \( e_\phi \) direction as:

\[
\min(\langle \dot{X}_T \rangle_{k+1}, \sqrt{V_{\text{lim} r}^2 - \langle \dot{X}_R \rangle_{k+1}^2}).
\]

Set the velocity limit as the sub-goal velocity and set

\[
A_{\text{lim} \phi} = \sqrt{A_{\text{lim} r}^2 - \langle \dot{X}_R \rangle_{k+1}^2}.
\]

(Step 4 is included to suit cases when the end-effector is moving with a high speed in a direction perpendicular to the target's velocity.)

5.4.3 Switching from Navigation to Tracking

The mod-QPT performs target tracking in a similar manner to IPNG, by trying to turn the relative velocity onto the LOS. However, although during the IPNG phase of motion, the end-effector's closing velocity is continually increased (by adding an acceleration component along the LOS), in the mod-QPT the end-effector's closing velocity is (continually) modified such that velocity matching along the LOS is ensured at the interception point.

When the mod-QPT takes over control from IPNG, it will not start to decelerate the end-effector immediately if the end-effector is still far from the target. Since the mod-QPT
performs target tracking in a similar fashion to IPNG, the interception time will not vary significantly with the switching instant, provided that late switching is not performed. A simple approach was, thus, adopted herein for the selection of the switching instant: A pre-defined distance threshold is used to trigger switching from IPNG motion-planning to the mod-QPT method. This approach is similar to that proposed in [21], used to switch from a coarse-tuning tracking method to fine-tuning method. A conservative distance threshold, which ensures that late switching does not occur, can be selected a priori.

5.4.4 Simulation Example

The interception of a fast-maneuvering target is presented in the following example. The target is moving on the same sinusoidal trajectory as defined by Equation 3.28 for the original simulation example of the hybrid IPNG-QPT scheme, Figure 3.6:

\[
X_T = \begin{bmatrix}
1400.0 + 30.0t \\
150.0 - 200.0 \sin\left(\frac{\pi}{4} t\right)
\end{bmatrix} \text{mm.} \tag{5.5}
\]

However, planar interception is considered herein, i.e., the end-effector is initially at rest in the target’s plane, at \([0 0 0]^T\) mm, rather than at \([0 0 150]^T\) mm. This corresponds to a negligible difference in the initial distance between the end-effector and the target, 7 mm. The tolerances of interception and the end-effector’s speed and acceleration limits are identical, i.e., \(tol_p = 10\) mm, \(tol_v = 10\) mm/s, \(V_{lim} = 300\) mm/s and \(A_{lim} = 400\) mm/s\(^2\).

Figure 5.4 shows that there is very little variation in the interception time as the switching time is varied. Figure 5.5a depicts the end-effector’s and target’s trajectories when switching from IPNG to mod-QPT at 5.7 s, yielding an interception time of 7.0 s. Figure 5.5b shows the end-effector’s trajectory when switching very early at 2.6 s. It can be seen from the plots that the paths of the end-effector are similar. The difference in the interception time is only one robot sampling interval, i.e., 0.22 s.

Figure 5.6 depicts the variations of the \(x\) and \(y\) components of the position and velocity of the target and the end-effector versus time when switching at 5.7 s.
Figure 5.4: Variation of $t_{int}$ versus switching time for a sinusoidal target trajectory.

Figure 5.5: End-effector's and target's trajectories: (a) switching at 5.7 s, and (b) at 2.6 s.
The mod-QPT method has been developed in this thesis for the hybrid IPNG-based interception scheme. Modifications are required for the mod-QPT method to be used as an independent method. In particular, a proper motion-initialization scheme must be incorporated into the method and the algorithm for dividing the velocity and acceleration limits must be modified, to ensure that, initially, the end-effector does not move in an unoptimal direction.

5.4.5 Experimental Results

The interception of a moving target using the hybrid IPNG-mod-QPT scheme was examined on the experimental test-bed. The same target trajectories presented in the examples of the original hybrid IPNG-QPT method were considered. The end-effector is initially located in the same plane as the target, at $[200 \ -300 \ 0]^T$ mm. In the experiments of Chapter 4, the end-effector was originally at $z = 50$ mm. This corresponds to a difference in initial distance between the target and the end-effector of approximately 3 mm. The tolerances of interception
were set at $t_{ol_p} = 10$ mm and $t_{ol_v} = 10$ mm/s, the same used in the examples of Chapter 4. The distance threshold at which switching is performed for the hybrid IPNG-mod-QPT method was set to 100 mm.

5.4.5.1 Slow-Maneuvering Targets

The end-effector's velocity and acceleration limits for both examples below were set as $70$ mm/s and $150$ mm/s$^2$ respectively.

Figure 5.7 depicts the interception of a target moving on a linear trajectory, (Figure 4.3), at approximately $30$ mm/s using the hybrid IPNG-mod-QPT method. Switching occurs at $7.0$ s, and the target is intercepted at $9.4$ s. The hybrid IPNG-QPT method yielded interception at $9.6$ s, (i.e., approximately one motion interval later).

![Figure 5.7: End-effector and target trajectories (linear target trajectory).](image)

Figure 5.8 shows the variation of the position of the end-effector and of the target versus time (when switching is performed at $7.0$ s). Figure 5.9 shows the variation of the interception time versus the switching time. The curve shows that the interception time is not very sensitive to the switching instant in the range of $0.5$ s to $6.0$ s.
Figure 5.8: Position variation versus time for the end-effector and the target.

Figure 5.9: Variation of the interception time versus the switching time.

Figure 5.10 illustrates the interception of a target moving on a circular trajectory, (Figure 4.6). The hybrid IPNG-mod-QPT method yields an interception time of 7.1 s, whereas the original hybrid IPNG-QPT method yielded an interception time of 7.6 s. Figure 5.11 shows that the interception time is not very sensitive to the switching instant in the range of 0.5 s to 6 s, for this case as well.
5.4.5.2 Fast-Maneuvering Targets

Figure 5.12 depicts the interception of a target which is initially moving on a circular trajectory and that subsequently rapidly reverses its direction, (Figure 4.9). Using the hybrid IPNG-mod-QPT method, interception occurs at 9.3 s. However, in this case, when using the original hybrid IPNG-QPT method, interception actually occurred one motion interval before, i.e., at 9.1 s. Figure 5.13 illustrates the variation of the end-effector's and target's positions versus time for the original hybrid IPNG-mod-QPT method.
Figure 5.12: Interception of a target moving on a circular trajectory and changing its direction of motion.

Figure 5.13: Position variation versus time for the end-effector and the target.

The interception of a target which initially moves on a linear trajectory and subsequently changes to a circular trajectory is illustrated in Figure 5.14. The hybrid IPNG-mod-QPT method yields an interception of 7.8 s. Using the original hybrid IPNG-QPT method, the target is intercepted at 8.4 s.
5.4.6 Discussion

The experimental results obtained indicate that, given the speed and acceleration superiority of the end-effector when compared to the target and given the interception tolerances utilized on the experimental testbed, the IPNG-mod-QPT scheme yields the following two primary advantages over the IPNG-QPT method: less sensitivity to the switching instant, and, generally, faster interception times.

The plots of the interception time versus the switching time indicate that if switching were to be performed when the target is still far from the target, very similar interception times would be obtained. This indicates that the mod-QPT succeeds in tracking the target in a similar manner to IPNG when the target is still far; namely, by trying to maintain a constant heading with the target and increasing the closing velocity. For all the experiments considered, the switching instant had no significant influence on the interception time (for a wide range of switching times).
Thus, the preliminary results indicate that the mod-QPT method may succeed in yielding an intercept course for the end-effector similar to the IPNG technique when the target is still far. However, no conclusive evidence may be drawn from the experimental results presented above, and numerous issues still need to be addressed within a theoretical framework, particularly the convergence criteria of the tracking technique utilized.

The mod-QPT method presented in the previous sections has been developed for examination purposes and requires improvements. The algorithm in which the velocity and acceleration limits are divided to enable the two trajectories to be generated independently must be optimized, (in particular the division of the acceleration limit). The process of mapping vectors using the rotational operator as defined in Section 5.4.2 has a singular point at $r = [0 0]^T$ (i.e., the end-effector and the target are currently co-located), and this has yet to be addressed. Moreover, the method should be extended to suit for three-dimensional interception.

5.5 Summary

In this chapter, the QPT has been modified to perform navigation-based tracking. The mod-QPT tries to maintain a constant heading with the moving target while closing the distance, in a similar fashion to IPNG. However, it ensures that the relative velocity is reduced to zero at the interception point. In the experimental results, the interception time was not very sensitive to the switching instant, indicating that the modified tracker may yield very similar interception times even if it takes control very early during the motion. Numerous issues, however, remain to be addressed within a theoretical framework.
Chapter 6: Conclusions

6.1 Summary & Conclusions

In this thesis, the implementation of an IPNG-based system for moving object interception has been presented. The primary goals of the thesis were: (1) the development of an IPNG-based robot-motion planner suitable for the GMF FANUC S-100 and its Karel controller, based on the method proposed by Mehrandezh et al. in [3]; and, (2) the implementation of the interception scheme in real-time.

The first goal was successfully achieved by developing new motion-planning algorithms that can servo the Karel controller with end-effector positional data. The interception scheme consists of a hybrid system in which a tracking technique takes over control of the manipulator from the IPNG technique, as originally proposed in [3]. However, significant changes were made within the framework of this thesis to suit the industrial robotic system in the CIMLab, whose dynamic model was not available and which has a closed controller.

During the IPNG phase of the motion, the end-effector's motion is controlled via the IPNG acceleration command. Subsequently, the acceleration command is modified subject to the manipulator's capabilities. Since the robot dynamics could not be taken into account, the acceleration command is modified subject to the worst case task-space velocity and acceleration capabilities of the GMF S-100. A constant acceleration model is utilized to generate the end-effector setpoints from the computed acceleration commands.

The Computed-Torque tracking method originally utilized in [3] was replaced in this thesis by a Quintic-Polynomial Tracking method. In this method, target tracking is achieved by generating a quintic-polynomial trajectory from the current end-effector state to the target's state. The motion time of the trajectory is minimized, subject to the velocity and acceleration constraints of the manipulator, to ensure fast interception of the target. A new switching methodology, which does not require long-term target-path prediction, was developed to enable the on-line selection of the switching instant in the hybrid scheme.
To achieve the second goal of the thesis, a PC-based real-time interception system was developed using the equipment available in the CIMLab. The motion-planning algorithms were implemented on a Pentium-CPU PC. The computation time of these algorithms is negligible compared to the vision system delay and the delay due to the Karel controller. A vision module was also developed for real-time target tracking. A Kalman Filter, developed earlier in our laboratory, is introduced in the system to provide target-state prediction. To enable the real-time, asynchronous functioning of the different components in the multi-architectural system, low-level PC-to-PC and PC-to-robot-controller communication interfaces were developed.

The overall IPNG-based interception system was thoroughly tested for a variety of target trajectories. The experimental results demonstrate that the hybrid IPNG-based interception scheme is capable of intercepting slow-maneuvering and fast-maneuvering targets faster than a (pure) tracking scheme.

During the implementation, standard industrial robotic equipment and personal computers were utilized. Thus, it was shown that the proposed interception scheme can be easily adopted in a manufacturing setting. Moreover, the modular approach adopted in designing the system enables the real-time experimentation of other motion-planning schema that may be developed in the future.

The QPT method was modified to assess the feasibility of performing navigation-based tracking using the GMFanuc S-100 manipulator. Preliminary experimental results indicate that this tracking method succeeds in tracking the target in a similar manner to a navigation-technique when the end-effector is still far from the target, while ensuring velocity matching at the interception point. However, no conclusive results could be drawn due to the limited time frame of this thesis.
6.2 Recommendations

The work conducted in this thesis can be extended in various directions. The recommendations discussed briefly below address two potential extensions of this work: (1) improvements on the experimental test-bed in the CIMLab, and, (2) the development of new motion-planning algorithms for moving-object interception.

(1) Improvements on the experimental system in the CIMLab.

(i) Throughout this thesis, interception was defined in terms of the magnitude of the relative position and velocity errors between the end-effector and the target. No effort was made to grasp the target when the end-effector is located at a pre-grasp situation. The grasping problem is complimentary to the interception task, and would render the interception scheme more useful for industrial applications. It must be acknowledged, however, that the grasping problem is a complex task involving numerous issues, such as object recognition and fine-motion planning via proximity-sensing (or eye-in-hand vision).

(ii) In this thesis, short-term target path prediction was achieved via a Kalman Filter developed earlier in our laboratory. The filter limits the response of the interception system to sudden target maneuvers. These effects should be investigated, and compared to the response of other types of prediction techniques suitable for short-term target path estimation.

(2) Development of new motion-planning strategies

(i) In Chapter 5, the feasibility of performing navigation-based tracking was examined for the GMFanuc S-100 robotic system in the CIMLab. The concept of navigation-based tracking should be studied within a theoretical framework, taking into account the robot’s dynamics. In particular, the convergence of such a technique (e.g., for different target maneuvers and size of the interception tolerance bands) should be examined, and the method compared to conventional tracking techniques, which take into account the robot’s dynamics, e.g., the Computed-Torque tracking method.
The preliminary results obtained from the modified Quintic-Polynomial Tracking method on the robotic system in the CMLab indicate that the method is superior to the original QPT within the hybrid IPNG-based interception system. Thus, the development of the technique as an independent tracking method is recommended. Various modifications are necessary to the algorithm presented in Chapter 5, in particular: the incorporation of a motion-initialization scheme, the optimization of the division of the acceleration and velocity limits, and the extension of the method to suit 3D interception.
References


