Adaptive Control of a Camera–Projection System using Vision–Based Feedback

Chwen Kai Liao

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Mechanical Engineering

Andrew J. Kurdila, Chair
Steve C. Southward, Co-Chair
Craig A. Woolsey

March 4^{th}, 2016
Blacksburg, Virginia

Keywords:
Adaptive control; nonlinear systems; vision based servo control; image projection systems.

Adaptive Control of a Camera–Projection System using Vision–Based Feedback

Chwen Kai Liao

(ABSTRACT)

This thesis derives an vision based feedback control strategy for a class of uncertain projector-camera systems that are used to animate two dimensional projected images on complex, three dimensional, articulated target objects. The target object of the robotic system is articulated using an open loop control strategy that generates a desired sequence of target poses that are designed using commercially available geometric modeling software. The ideal or desired image sequences are subsequently rendered in the geometric modeling software using an ideal camera/projector pose and ideal intrinsic parameter camera model. The rendered imagery from the ideal camera and projector pose are subsequently used to define tracking performance for the feedback control of the camera and projector. Uncertainty in actuator models of the camera and projector actuator subsystems in this paper includes contributions due to imprecision in camera pose and in intrinsic camera parameters. A feedback control strategy is derived that employs pixel coordinates of multiple tracked feature points in the target image sequence for pose estimation and tracking control problems. We establish sufficient conditions that guarantee the convergence and asymptotic stability of the pose estimation and tracking control problems for the class of uncertain, nonlinear systems studied in this thesis. Several numerical studies are summarized in the thesis that provide confidence in the derived theoretical results and further suggest robustness of the control strategy for the considered uncertainty class.

Adaptive Control of a Camera–Projection System using Vision–Based Feedback

Chwen Kai Liao

(GENERAL AUDIENCE ABSTRACT)

Over the past few years, people have tried to project the image onto certain 3 dimensional objects in order to make the projection image more vivid. However, most of the projection system designs belong to static projection, which means the projector remains static when it projects the image. The potential limitation of static projection scenario is the restricted projection area, which means the projection target can only stay within a fixed projection area or the image can not be projected onto the target. In order to release this limitation, we propsed a mobile projection solution so that the porjector can move in the space and track the target. By this design, the valid projection area is no longer fixed but can move with the target due to the advantage of mobile projection design.

Contents

List of Figures

List of Tables

Nomenclature

C	Camera frame designation
$(X_{C,i}, Y_{C,i}, Z_{C,i})$	Coordinate of point i with respect to camera C frame
P	Projector frame designation
C_i	Controlled robot body frame $(i = 0, 1, 2, ...L)$
P_i	Projector robot arm frame $(i = 0, 1, 2, ...L)$
T	Projection target designation
Ω	Camera pixel coordinate
f_i^*	Point i on projection target T
γ_i	Point i on the projected image Γ
Γ	Image projected from the projector P during projection tracking
$\widetilde{C}$	Perfect camera mounting coordinate on the robot
P^*	Desired projector frame
F	Arbitrary Cartesian frame F
I	Inertial frame
$(\hat{\boldsymbol{x}}_F, \hat{\boldsymbol{y}}_F, \hat{\boldsymbol{z}}_F)$	Basis vector of arbitrary Cartesian frame F
(u_i, v_i)	Coordinate of point i in camera canonical retinal coordinate frame Ω
$\boldsymbol{r}_{K,a}^F$	Vector from point a to point b expressed in frame F
$\boldsymbol{v}_{A,p}^F$	Vector of point p relative to frame A expressed in frame F
$\boldsymbol{\omega}_{A,B}^F$	Angular velocity vector $\boldsymbol{\omega}$ of frame B relative to frame A expressed in frame F
$\boldsymbol{H}_A^B$	From the frame A to the frame B
$\boldsymbol{s^*}$	Desired image feature in vector form
$\boldsymbol{s}$	Image feature in vector form
$\boldsymbol{A}^\dagger$	Pseudoinverse of matrix A
F_i	Vector of the force in Cartesian space acting on the particle i

Chapter 1

Introduction

Over the past few years, artists working in collaboration with engineers and computer scientists have made impressive progress in utilizing computer graphics animation techniques, video projection methods, and camera/projector control strategies to generate stunning visual displays. The production of these graphical displays requires the synthesis of techniques of photogrammetry, video animation, and computer control.

At its root, these methods rely on the careful generation of rendered video images that must be projected onto a moving three dimensional object. Because the end desire is to produce photo-realistic imagery, the production of the video script that is projected or rendered onto the surface of the moving target can be a computationally expensive task, one that depends on the complexity of the desired video. It is both a substantial endeavor to design such a video script as well as a significant computational task to render it. Moreover, as the dynamic content of the video script becomes more complex, it is critical that the characteristics of the physical camera and projection system match as closely as possible those of the virtual system that is used to render the video template. A mismatch in the extrinsic camera parameters (i.e., the location and orientation of the imaging system relative to the target robotic arm) or in the intrinsic camera parameters can degrade the final projected display. In such cases the final video display can be unconvincing and may not produce the intended three dimensional effects.

One common approach to control the servo motor of a projector system is to introduce the vision based feedback from a camera. In general, the vision based servo control strategy can be separated into two main categories: position based visual servoing *(PBVS)* and image based visual servoing *(IBVS)*.

When applying the position based vision servo control method, it is necessary to estimate the pose of the projection target with respect to the camera based on the camera image measurement. The obvious drawback here is that the estimation pose of the projection target can be significantly affected by an inaccurate camera calibration. Pre-knowledge of

Figure 1.1: 3D projection mapping by Bot & Dolly company

Figure 1.2: Other projection mapping work. *(https://vimeo.com/50197298)*

the projection target geometry is required so that the 3D reconstruction of the target pose can be more easily computed from the image measurement of the camera. Also, since the control law does not incorporate any constraint in image plane coordinates, it is possible

for the target to leave the field of view of camera. This can cause the controller to fail. In addition, the error between the current and the desired camera-projector system pose is defined in the task space, which is given in Cartesian coordinate frame. Expressing the error term in Cartesian coordinates is advantageous in that motion planning can be easier to compute. The design of the system end effector trajectory can be conducted in three dimensinal space to deal with object avoidance. The synthesis of a controller of position based visual servo control achieves convergence property in Cartesian space.

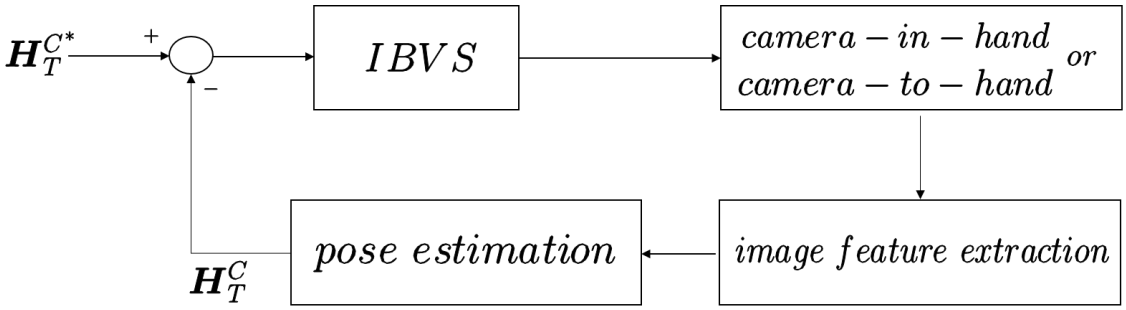


Figure 1.3: Position based vision servo

On the other hand, instead of a controller to seek convergence in Cartesian space, the image based visual servo control *(IBVS)* method establishes convergence in camera image coordinates. To apply the *IBVS* in the camera-projector system, one of the challenges is to derive the *interaction matrix* or the *image Jacobian*. This matrix which maps the target velocity from the Cartesian space to image plane coordinates. In the camera image plane, it is required to select the image features on the projection target for the controller to track. The image feature for the projection target can be specific points, or it can be some generic representation of the image such as image moments. During the formation of the interaction matrix, any camera image plane constraints can be easily introduced into the controller design. This can be an adventage in camparison to a position based visual servo strategy. The proposed image plane constraints can prevent the image of the projection target from running out of the field of view.

In addition, unlike *PBVS*, the image based visual servo control does not require the estimation of the target geometry. This is advantage makes *IBVS* applicable to the case where the projection target has unknown geometry. In addition, The only Cartesian space information needed is the range from the camera to the target. One of the common methods to estimate this range installs multiple cameras in the projection system and generates stereo vision. The stereo vision is a technique of reconstruction from two images taken from different viewpoints. The range information can be calculated from the 3D reconstruction in real time. Due to these adventages, the *IBVS* is considered to be very robust with respect to camera and projection system method calibration error.

In the thesis, the image based visual servo control method is selected because the geometry of the projection target is not previsously known, and also the projection system end effector

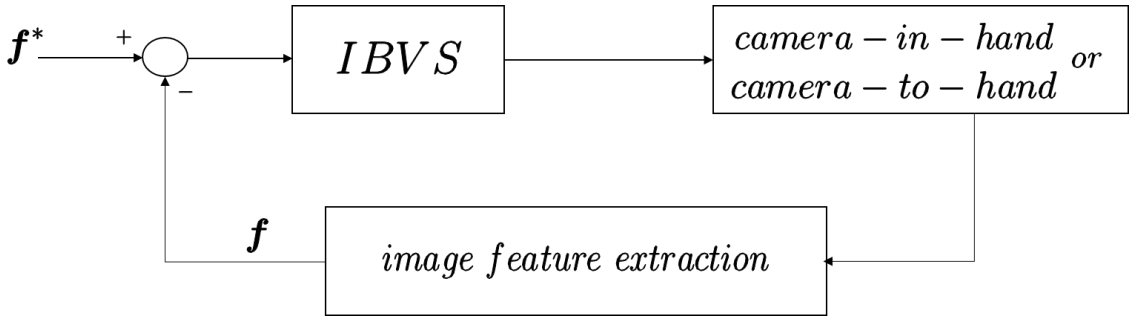


Figure 1.4: Image based vision servo

is not required to perform obstacle avoidance. Unlike other projection mapping system designs where the camera and projector remain static, the design in our study employs a mobile camera-projector unit. The advantage of our design is that the projection region is much wider than the static case. In our study, the camera and the projector are rigidly fixed together on the robot end effector, so it constitutes a classical to eye-in-hand IBVS control problem. The robot arm be used in the study is ABB IRB120 shown in Figure 1.5 from ABB company, and the camera-projector unit is driven by the robot manipulator.

Figure 1.5: ABB IRB120 robot, from ABB company

Because the camera and projector are installed on the robot end effector by the user, the mounting error of this unit can be significant with respect to other errors that are caused by the manufacturer. To tackle the mounting error, we propose a method that combines an adaptive control strategy within the image based visual servo control framework. This

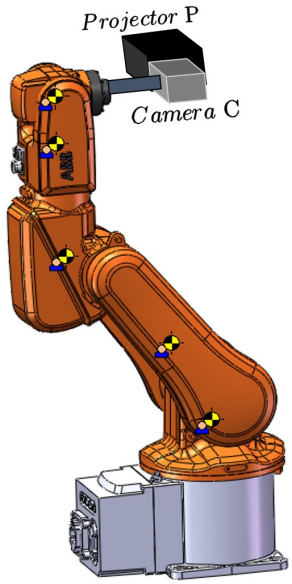


Figure 1.6: The projector and the camera are rigidly fixed on the robot end effector

strategy can effectively deal with the uncertain orientation of the camera-projector unit. In addition, if the camera calibration was performed poorly, the camera intrinsic parameters may contain error. The large intrinsic parameter error can lead to performance failure if no error compensation is introduced in the controller. In order to resolve the uncertainty in camera intrinsic parameters issue, the adaptive controller we derive in the later chapter will include the camera intrinsic properties, which enables the treatment of poor camera calibration.

In Chapter 2, we will illustrate how the control problem is formulated and the goals of the controller. Next, in Chapter 3, we discuss the kinematic and dynamic models of the robot manipulator and the camera. In addition, Chapter 4 presents a treatment of the uncertainties that appear in our model. Furthermore, Chapter 5 introduces the adaptive controller and shows that it can compensate those modeling errors mentioned in Chapter four. In Chapter 6, we perform case studies of different combinations of the uncertainty and study the system performance.

Chapter 2

Problem description

In this thesis, we study a system that is constructed from two robotic arms as depicted in Figure 2.1. The large robot arm holds the prjection target. The projection target is denoted as T. In addition, the large robot arm is operated in open loop control mode, which means we have no ability to control the large robot by feedback. The projection target T moves along a predefined trajectory in the inertial space.

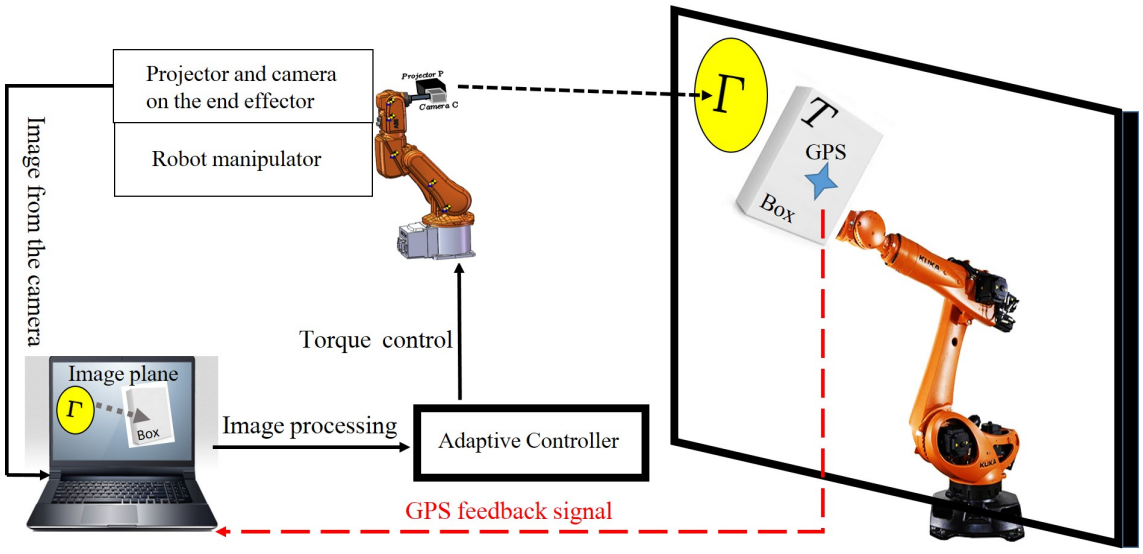


Figure 2.1: The overview of the problem setup

On the end effector of the small robot arm (*ABB IRB120*) ; the camera and projector are rigidly fix together. The projector P outputs the image Γ which is previously generated and rendered by artists. The camera C is used for measurements for the vision based feedback controller in an image based servo control strategy. The image captured by the camera C contains projected image Γ and projection target T. The image is subject to image processing

to extract features. The image feature is defined to be some characteristic of the image. It can be the certain selected points on the image or a generic property of the image such as the center of the image or image moments. With selecting one type of the image feature, we calculate the image features for the projected image and the projection target. The image feature of the projected image is denoted as $\boldsymbol{s}$, and the projection target image feature is called desired image feature $\boldsymbol{s}^*$. We want the projected image Γ to be projected on top of the projection target T.

After the image feature extraction has been done, the controller uses the extreacted image feature to compute the required torque on each ABB IRB120 robot actuator so that $\boldsymbol{s}$ converges to $\boldsymbol{s}^*$ in camera image plane. In addition, the image base visual servo control method requires the range between the camera C and the projection target P. Because we only have one camera installed on the robot end effector, using the stereo vision to predict the range is not possible since the method needs at least two cameras. To deal with this issue, the GPS sensor is installed inside the projection target T to measure the range in real time.

2.1 Image rendering space

Before the control strategy is executed, it is necessary that artists render the projection image in a simulation software package such as MAYA®. In this type of simulation software, the user needs to assign the projector location and its orientation relative to the projection target as Figure 2.2 shows. After the projector pose is defined, the user can then begin working on moving the objects in image rendering. Using MAYA® as an example, the images for each projector are generated from the scene in the simulation environment. That is, MAYA® form a two dimensional image from the specific view of the three dimensional scene. The specific views are defined from choices where the user assigns the projectors in the simulation environment. After the preliminary rendered images for each projector are generated, the user can further tune the image properties in order to achieve better image effects on the projection target.

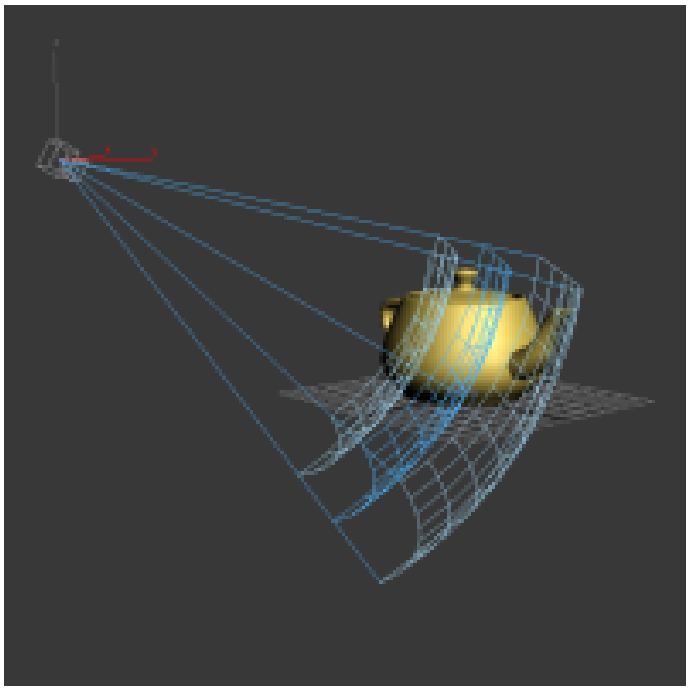


Figure 2.2: Define projector pose in MAYA®.*(https://www.vray.com)*

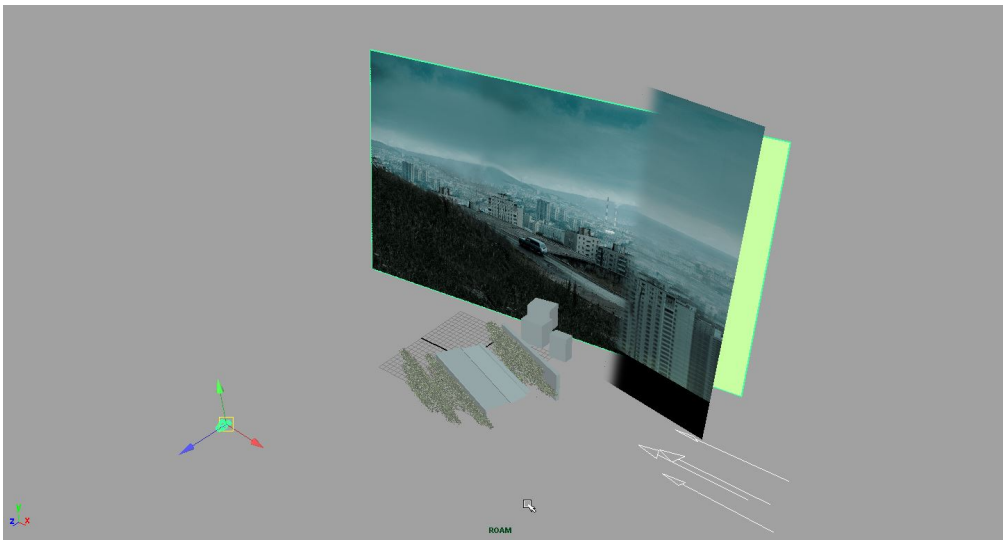


Figure 2.3: Image projection examples. *(http://www.creativebloq.com/)*

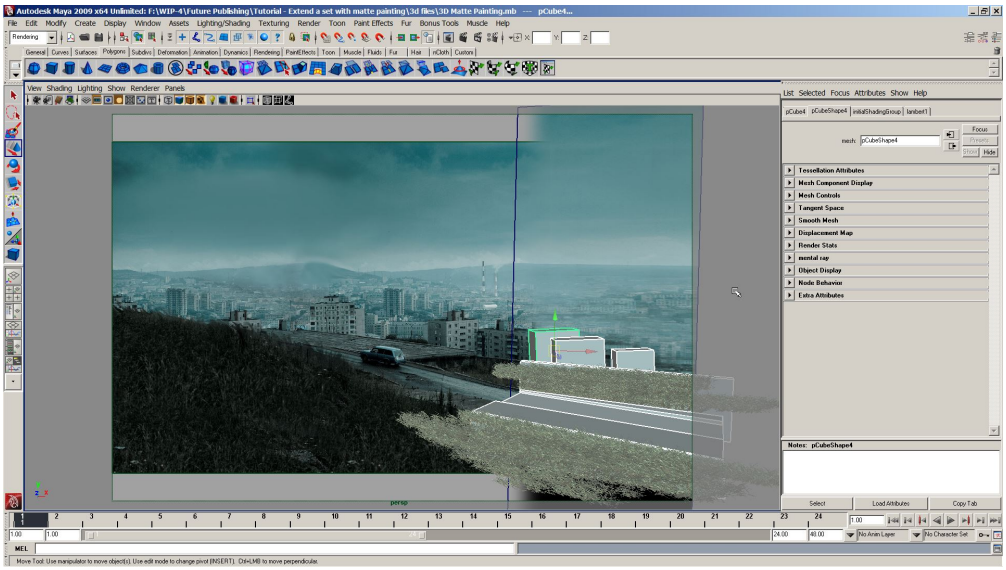


Figure 2.4: Example projected image collage in MAYA®. *(http://www.creativebloq.com/)*

Figure 2.2 shows the rendering space of MAYA®. The relative pose between the projector and the projection target need to be defined before starting making the projection image for the projector. The top left is a projector and the pot is the projection target. In addition, Figure 2.3 shows that in the simulation environment, there are 3 dimensional projection targets and 2 dimensional projected pictures. Furthermore, Figure 2.4 demonstrates the way to collage the projected pictures with projection targets.

2.2 Calibration

As noted in our introduction, the goal of this thesis is the faithful presentation of artistically designed animation scripts that are projected onto general three dimensional targets. It is assumed that prior to any experiment or display of the video script that the designers have carefully computed the rendering to be projected onto the target, and this step was discussed in the last section. This rendered sequence of images is computed from a fixed, ideal pose of the camera and projector location relative to the robot that actuates the target. When the display system is initially set up, the actual pose of the robot holding the camera and projector subsystem may not coincide with the ideal pose used to generate the rendering. This mismatch can result in substantial degradation of the intended visual effects and must be attenuated or minimized.

The first problem addressed in this paper is the development of an automated calibration process that accounts, insofar as it is feasible, for pose uncertainty. The camera image plane is selected as the task space because of the image based visual servo control strategy we used.

In addition, we extract the image feature from the image of the current projection target and denote the image feature as $\boldsymbol{s}$. Next, the desired porjection target location is defined in the camera image plane and its corresponding image feature is denoted as $\boldsymbol{s}^*$. The calibration process is intended to compensate for the relative pose between the camera C and the static projection target T by driving the image feature $\boldsymbol{s}$ to match $\boldsymbol{s}^*$ in camera pixel space Ω. After a successful calibration is complete, the configuration of the projection system and the projection target are the same as that in the rendering software. Note that during the calibration, the projector does not project the image, and only the robot manipulator and the camera C is actuated. The calibration process is illustrated in Figure 2.6.

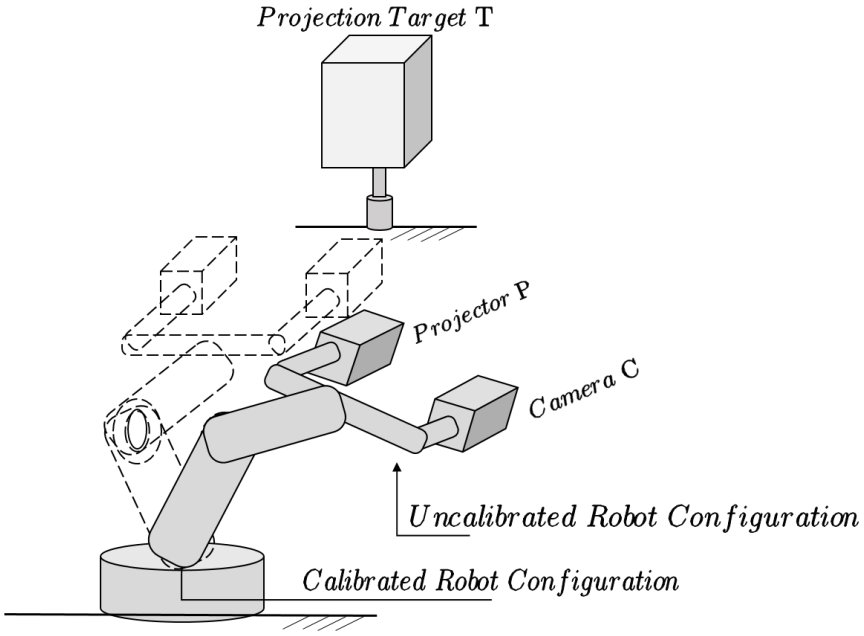


Figure 2.5: Robot manipulator configuration during the calibration porcess

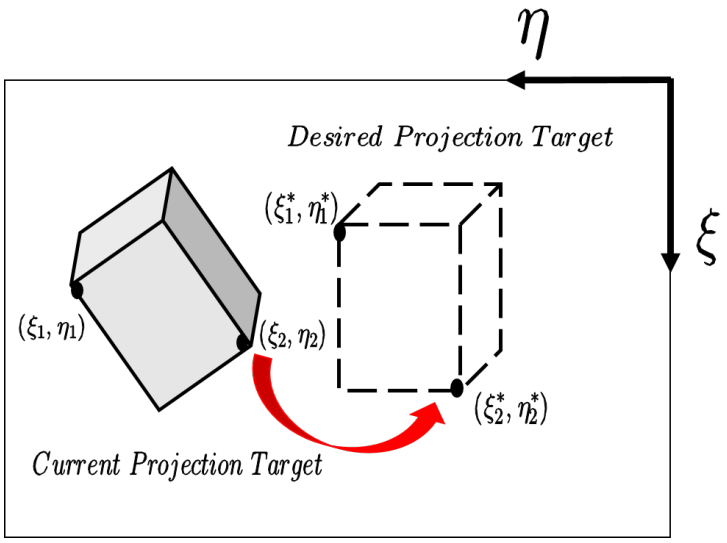


Figure 2.6: The view of the camera during the calibration process

2.3 Target tracing projection

After the successful calibration of the robot manipulator is computed, the camera-projector unit and the projection target pose are the same as the pose that artists defined in the image rendering space. The system is now ready to perform target tracing with projecting the rendered image onto the target. In the later chpater we will derieve the controller for target tracing projection, and give out the stability proof of the control system.

To study the target tracing projection problem, the camera image plane is selected to be task space. The goal of the projection target is to be tracked so its image features coincide with $\boldsymbol{s}^*$. In addition, the image of the projected image Γ in camera image plane is selected as the control object, and it's image feature is denoted as $\boldsymbol{s}$. The ultimate goal is to drive the image feature $\boldsymbol{s}$ of the projected image Γ to match the image feature $\boldsymbol{s}^*$ of the projection target T. Becasue the projected image has been pre-wrapped by the artists in the rendering software, the projected image will perform well once Γ is projected on the correct location on the projection target T. To guarantee the projected image Γ is projected correctly and moves along with the target, the controller we derive should be able to perform tracking. In addition, since the camera-projector unit may have mounting error caused by the user and camera intrinsic uncertainty due to poor camera calibration, the adaptive control strategy is designed to deal with this issues.

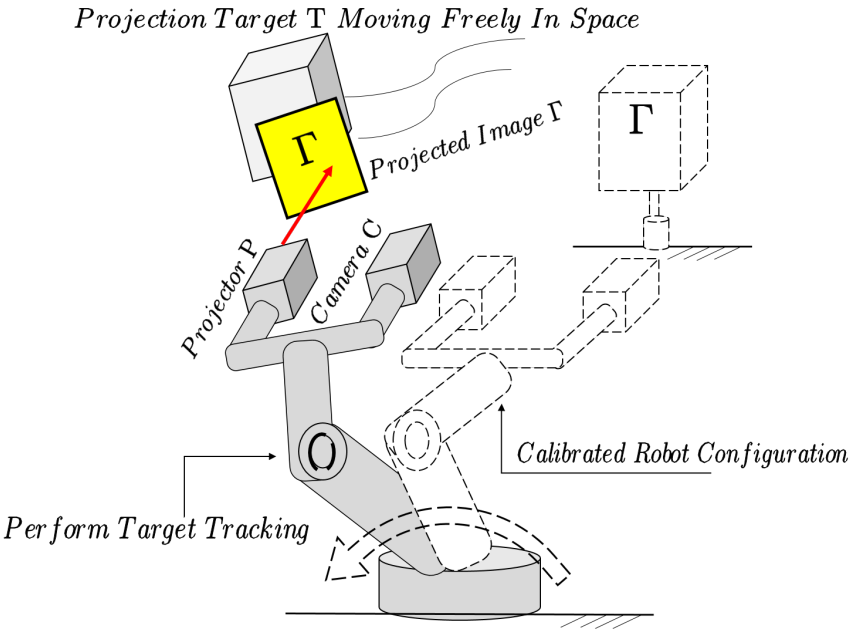


Figure 2.7: Robot manipulator configuration during the target tracing projection porcess

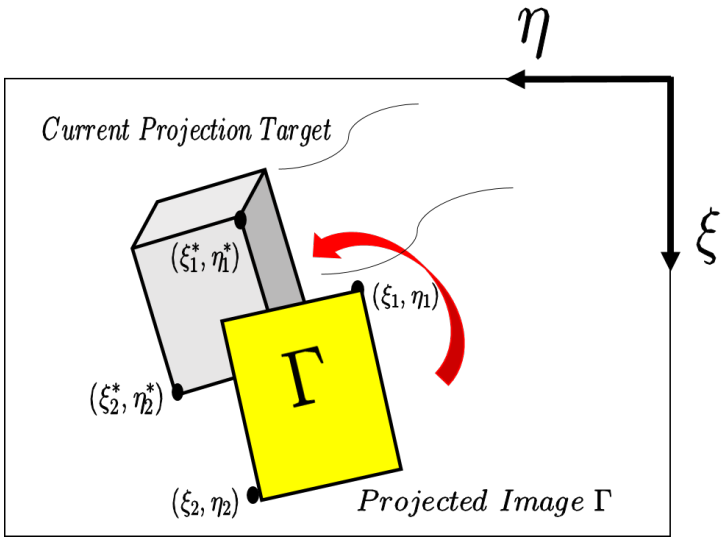


Figure 2.8: The view of the camera during the target tracing projection process

2.4 Coordinate systems

The general configuration of the camera and projector system is depicted in Figure 2.10. Following the Denavit-Hartenberg Convention for serial robotics , the frames of the kinematic chain are numbered beginning at the root and proceeding to the end effector.

For the camera robotic arm, each body-fixed frame is denoted C_i $(i = 0, 1, 2...L)$, and the corresponding joint angles are q_{C_i} $(i = 1, 2, ...L)$. As denoted in the figure, the relative position vector $\boldsymbol{r}_{F,G}$ connects the origin of the frame F to the origin of frame G. For example, the position vector of frame C_{i+1} relative to frame C_i is denoted $\boldsymbol{r}_{C_i,C_{i+1}}$, while the position vector of frame T relative to the inertial frame I is $\boldsymbol{r}_{I,T}$. As shown in Figure 2.10, the feature points f_i^* for $(i = 1, \ldots, n)$ are fixed on the target object T, and the position vector of feature point f_i^* in the camera frame C is given by $\boldsymbol{r}_{C,f_i^*}$.

The choice of coordinates or basis used to represent any vector quantity are indicated using superscripts. Therefore, the coordinates relative to a basis for the I frame of the position vector $\boldsymbol{r}_{I,T}$ are given by $\boldsymbol{r}_{I,T}^I$, while the components relative to the camera frame of the position vector $\boldsymbol{r}_{C,f_i^*}$ are denoted $\boldsymbol{r}_{C,f_i^*}^C$. The rotation matrix $\boldsymbol{R}_F^G$ achieves the change of variables from any frame F to frame G via the identity $\boldsymbol{a}^G = \boldsymbol{R}_F^G \boldsymbol{a}^F$ where $\boldsymbol{a}$ is an arbitrary vector and $\boldsymbol{a}^F, \boldsymbol{a}^G$ are its components relative to the F and G frames, respectively. In the discussions that follow, the 4×4 matrix $\boldsymbol{H}_G^F$ denotes the homogeneous transformation that represents the rigid body motion from the F frame to the G frame. We always have

$$\boldsymbol{H}_G^F := \begin{bmatrix} \boldsymbol{R}_G^F & \boldsymbol{r}_{F,G}^F \\ \boldsymbol{0}^T & 1 \end{bmatrix}. \tag{2.1}$$

The collection of points f_i^* $(i = 1, 2, ..., n)$ that are physically attached to the target T have the corresponding points γ_i $(i = 1, 2, ...n)$ in the projected image Γ. In our discussion that follows, the task space is understood to refer to the camera pixel coordinates in Ω. The camera C records the motion of T and Γ. It follows that the physical points on the projection target f_i^* $(i = 1, 2, ..., n)$ and selected points on the projected image γ_i $(i = 1, 2, ...n)$ will have pre-images that occur in Ω. The vector that points from the origin of Ω to the pre-image of f_i^* is denoted as $\boldsymbol{\phi}_{\Omega,f_i^*}^{\Omega}$. Likewise, The vector points that points from the origin of Ω to the pre-image of γ_i^* is denoted as $\boldsymbol{\phi}_{\Omega,\gamma_i^*}^{\Omega}$. Since the camera C and projector P are rigidly fixed with each other on the end of the robot arm, the vector $\boldsymbol{r}_{C_L,C}^{C_L}$ and $\boldsymbol{r}_{C_L,P}^{C_L}$ are constant vectors.

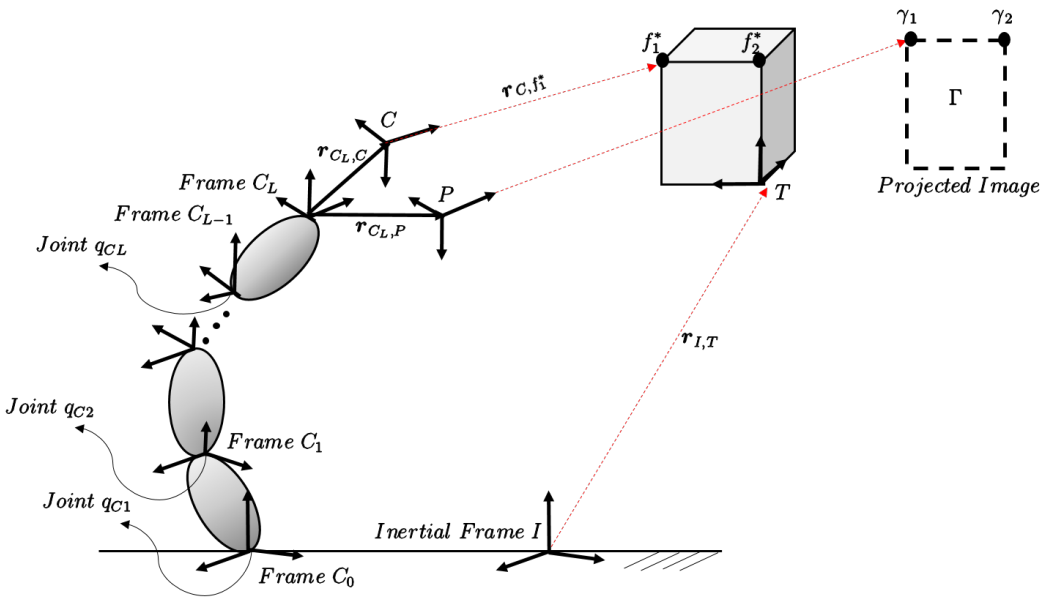


Figure 2.9: Frame and coordinate definitions for camera, projector, and target

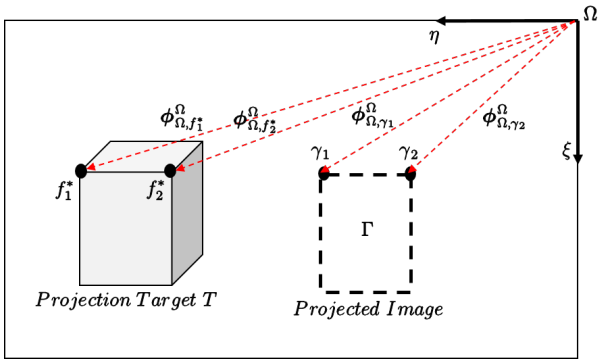


Figure 2.10: Graphical explanation of notations in nomenclature

2.5 Image feature extraction

In our problem, the camera pixel coordinates in Ω of the camera C parameterize the task space, and images of the points on projection target f_i^* $(i = 1, 2..., n)$ have position vectors $\boldsymbol{\phi}^{\Omega}_{\Omega,f_i^*}$ $(i = 1, 2..., n)$ relative to camera pixel coordinates in Ω. Also, the preimages are points γ_i $(i = 1, 2..., n)$ in the projection image Γ and their corresponding position vectors in canonical retinal coordinates Ω are $\boldsymbol{\phi}^{\Omega}_{\Omega,\gamma_i^*}$ $(i = 1, 2..., n)$. In our derivation of the control strategies, it will be convenient to assemble all the individual feature points in vector form. The assembly of all the position vectors are collected in the projection target $\boldsymbol{\Phi}^*$ and projected image

$\boldsymbol{\Phi}$ and can be written as

$$\boldsymbol{\Phi^*} = \begin{bmatrix} \boldsymbol{\phi}^{\Omega}_{\Omega,f_1^*} \\ \boldsymbol{\phi}^{\Omega}_{\Omega,f_2^*} \\ \vdots \\ \boldsymbol{\phi}^{\Omega}_{\Omega,f_n^*} \end{bmatrix}, \qquad \boldsymbol{\Phi} = \begin{bmatrix} \boldsymbol{\phi}^{\Omega}_{\Omega,\gamma_1} \\ \boldsymbol{\phi}^{\Omega}_{\Omega,\gamma_2} \\ \vdots \\ \boldsymbol{\phi}^{\Omega}_{\Omega,\gamma_n} \end{bmatrix}. \tag{2.2}$$

We define a possibly nonlinear function $\boldsymbol{h} = [h_1, h_2, \ldots, h_n]^T$ to map the collection of pixel coordinates $\boldsymbol{\Phi}$ to the task space vector $\boldsymbol{s}$, and to map the desired image feature $\boldsymbol{\Phi}^*$ to the desired task space variables $\boldsymbol{s}^*$.

$$\boldsymbol{s^*} = \begin{bmatrix} h_1\left(\boldsymbol{\Phi^*}\right) \\ h_2\left(\boldsymbol{\Phi^*}\right) \\ \vdots \\ h_n\left(\boldsymbol{\Phi^*}\right) \end{bmatrix} \qquad \boldsymbol{s} = \begin{bmatrix} h_1\left(\boldsymbol{\Phi}\right) \\ h_2\left(\boldsymbol{\Phi}\right) \\ \vdots \\ h_n\left(\boldsymbol{\Phi}\right) \end{bmatrix} \tag{2.3}$$

In the simplest case, the function $\boldsymbol{h}$ can just be an identity mapping, which physically means that the task space variables are exactly the images of selected feature points. In our case, when $\boldsymbol{h}$ is an identity mapping, we have

$$\boldsymbol{s^*} = \left[\begin{array}{c} \xi_{f_1^*} \\ \eta_{f_1^*} \\ \hline \xi_{f_2^*} \\ \eta_{f_2^*} \\ \hline \vdots \\ \hline \xi_{f_n^*} \\ \eta_{f_n^*} \end{array}\right] = h_1\left(\boldsymbol{\Phi^*}\right) = \begin{bmatrix} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ & & & & \ddots & & \\ & & & & & 1 & \\ & & & & & & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{\phi}^{\Omega}_{\Omega,f_1^*} \\ \boldsymbol{\phi}^{\Omega}_{\Omega,f_2^*} \\ \vdots \\ \boldsymbol{\phi}^{\Omega}_{\Omega,f_n^*} \end{bmatrix},$$

$$\boldsymbol{s} = \left[\begin{array}{c} \xi_{\gamma_1} \\ \eta_{\gamma_1} \\ \hline \xi_{\gamma_2} \\ \eta_{\gamma_2} \\ \hline \vdots \\ \hline \xi_{\gamma_n} \\ \eta_{\gamma_n} \end{array}\right] = h_1\left(\boldsymbol{\Phi}\right) = \begin{bmatrix} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ & & & & \ddots & & \\ & & & & & 1 & \\ & & & & & & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{\phi}^{\Omega}_{\Omega,\gamma_1} \\ \boldsymbol{\phi}^{\Omega}_{\Omega,\gamma_2} \\ \vdots \\ \boldsymbol{\phi}^{\Omega}_{\Omega,\gamma_n} \end{bmatrix}.$$

Or for instance, if the function h_1 is defined as $h_1 = \frac{\sum_{i=1}^{n} \boldsymbol{\phi}^{\Omega}_{\Omega,f_i^*}}{n}$, then the desired image feature vector $\boldsymbol{s^*}$ is the centroid $\left(\xi_{f_c^*}, \eta_{f_c^*}\right)$ of projection target points group f_i^* $(i = 1, 2, 3...n)$ in camera pixels in Ω. Likewise, the image feature vector $\boldsymbol{s}$ is the centroid $\left(\xi_{\gamma_c^*}, \eta_{\gamma_c^*}\right)$ of projected image points group γ_i $(i = 1, 2, 3...n)$. We then have

$$\boldsymbol{s^*} = \begin{bmatrix} \xi_{f_c^*} \\ \eta_{f_c^*} \end{bmatrix} = h_1\left(\boldsymbol{\Phi^*}\right) = \frac{\sum_{i=1}^{n} \boldsymbol{\phi}^{\Omega}_{\Omega,f_i^*}}{n},$$

$$\boldsymbol{s} = \begin{bmatrix} \xi_{\gamma_c} \\ \eta_{\gamma_c} \end{bmatrix} = h_1\left(\boldsymbol{\Phi}\right) = \frac{\sum_{i=1}^{n} \boldsymbol{\phi}^{\Omega}_{\Omega,\gamma_i}}{n}.$$

As for the image based servo control, different types of the image feature will have their it's own form of interaction matrix that describes the differential relationship between Cartesian velocity and the velocity in camera image space. Users that apply *IBVS* control methods need to derive the interaction matrix in advance.

Chapter 3

Robot modeling

In this chapter, we study the kinematic and dynamic modeling of the robot manipulator when the camera-projector unit is mounted on the end effector. We refer to the geometry specification that is available on the ABB® company official website when modeling the robot arm. In addition, the robot parameters such as mass and inertia of each robot link are obtained from the ABB IRB120 Solidwork® CAD model, which is also accessible to the public.

3.1 Robot kinematics

The kinematic of the robot models the geometry of the manipulator configuration without considering the mass, inertia, and the forces that are applied to the robot. The ABB IRB120 robot in our study is a twpe of the serial link manipulator robot arm, which means the manipulator consists of links and joints. To describe the motion of the robot arm, we use Denavit-Hartenberg parameter *(D-H parameeter)* to mathematically model the robot kinematics.

3.1.1 Denavit-Hartenberg parameter

In the Denavit-Hartenberg convention, the serial manipulator consists of links and two neighboring links bodies are connected by a joint. The type of the joint can incorporate either rotational or translational constraints, or both. The conventional way to number an N-body manipulator starts from number 0. Link 0 is assigned to the base of the manipulator. The joint that attaches the base is joint number 1. In the same way, the numbering proceeds outward along all the way to the robot end effector, which is numbered N.

The links in our study are considered to be a rigid bodies, and **joint j drives link j**.

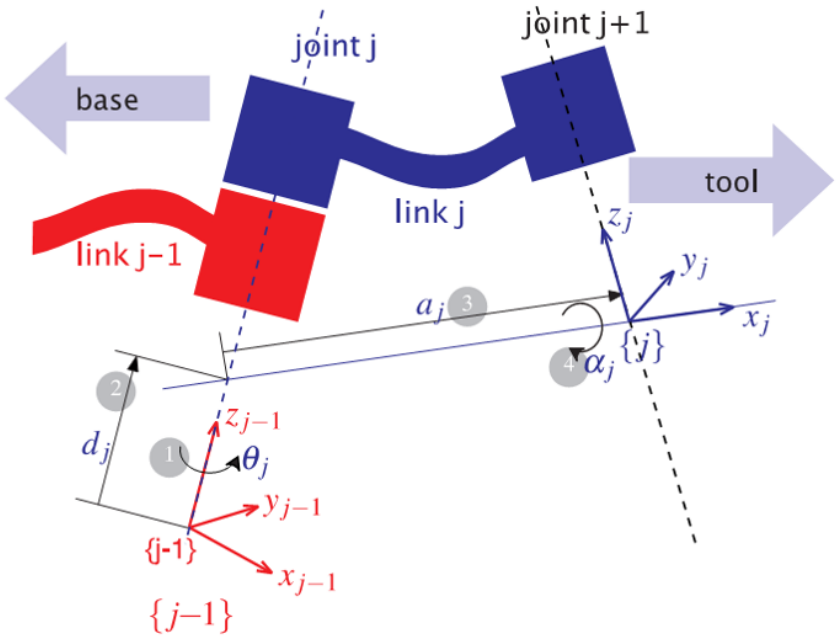


Figure 3.1: Definition of standard Denavit and Hartenberg link parameters. *(Robotics, vision ,and control. Peter Corke)*

In the standard Denavit-Hartenberg convention, four types of parameters are used: joint rotation, link offset, link displacement, and link twist. All of these parameters are defined the coordinates depicted in Figure 3.1. For the joint variable number j, the joint rotation is the angle between the x_{j-1} and x_j axis about the z_{j-1} axis, and we denoted the joint rotation number j as θ_j. The link offset d_j of joint variable j is the distance from the origin of the frame j-1 to the x_j axis along the z_{j-1} axis. In addition, the twist angle α_j of the joint variable j is the angle between z_{j-1} and z_j axis about the x_j axis. Finally, the displacement of the joint variable j is measuremed z_{j-1} and z_j axis along the x_j axis.

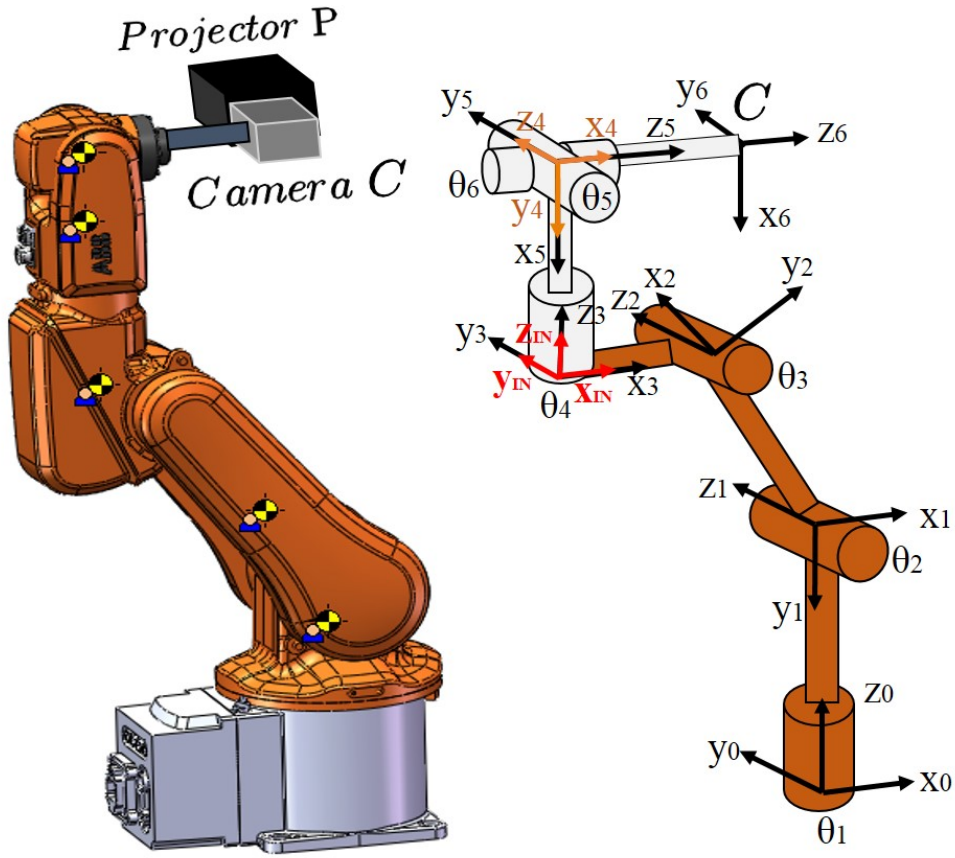


Figure 3.2: Robot body frame assignment on ABB IRB120 robot arm.

Table 3.1: Summary of Denavit-Hartenberg parameters for projector-camera robotic system

Link	$\theta_j(rad)$	$\alpha_j(rad)$	$a_j(mm)$	$d_j(mm)$
1	$\theta_1\,(t)$	$\frac{-\pi}{2}$	290	0
2	$\theta_2\,(t) - \frac{11\pi}{18}$	0	0	270
3	$\theta_3\,(t) + \frac{11\pi}{18}$	$\frac{\pi}{2}$	0	-70
4	$\theta_4\,(t)$	$\frac{-\pi}{2}$	302	0
5	$\theta_5\,(t)$	$\frac{\pi}{2}$	0	0
6	$\theta_6\,(t)$	α	d	δ

In our study, the ABB IRB120 body frames are assigned as depicted in Figure 3.2 shows. The robot arm has six degrees of freedom, starting from joint number 1 to joint number 6. The body frames are numbered starting from the base , which is denoted the 0 frame. The frame number 6 coincides with the camera frame, which is mounted on the robot end

effector. The joints number 4, 5, and 6 comprise the spherical wrist structure. The rotation axes intersect at one point. Frame number 4 and Frame number 5 are aligned so that they share a common origin.

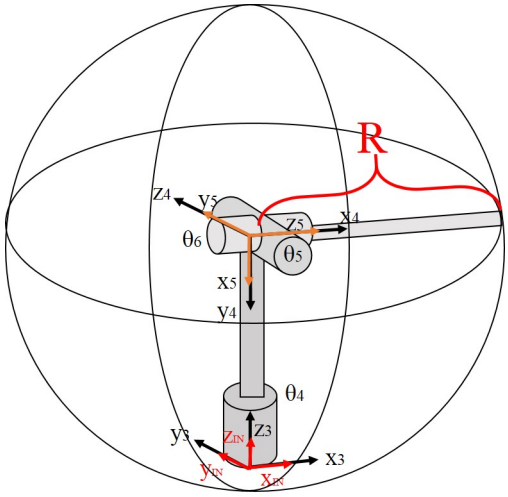


Figure 3.3: If not considering the collision, the reachable set for the spherical joint is the sphere with the radius equals to the length of the last linkage.

Although the base frame is selected as frame number 0, the inertial frame can be assign anywhere and does not necessarily need to coincide with the base frame. In fact, since in our study we only drive the last three joints of the robot, that is, joint numbers 4, 5, and 6, placing the inertial frame 3 is more adventageous. The inertial frame is referred to as (x_{IN}, y_{IN}, z_{IN}) in our study. As Figure 3.2 shows, the initial configuration is defined as $\left[\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6\right]^\intercal = \left[0, \frac{-11\pi}{18}, \frac{11\pi}{18}, 0, 0, 0\right]^\intercal$.

After the D-H parameters of the robot are defined, we can construct the homogeneous transformation that is used to describe the robot configuration. In the next section,we will discuss about how to build the homogeneous transformations using Denavit-Hartenberg parameters.

3.1.2 Homogeneous transformation

For the robot with body frames numbered from j=0,1,...N that all satisfy the D-H convention, it is possible to derive the homogeneous transformation one frame to another. The homogeneous transforamtion is constructed from a rotation matrix R_j^{j-1} that represents the relative orientation between the body frame j and j-1. In addition, the transformation also contains the displacement vector $\boldsymbol{d}_j^{j-1}$ that measures the offset between the origin of frame j and j-1. The homogeneous matrix has the form

$$\boldsymbol{H}_{j}^{j-1} = \begin{bmatrix} \boldsymbol{R}_{j}^{j-1} & \boldsymbol{d}_{j}^{j-1} \\ \mathbf{0} & 1 \end{bmatrix}_{4x4} \tag{3.1}$$

that maps from j to $j-1$. Let $\boldsymbol{a_x}, \boldsymbol{a_y}, \boldsymbol{a_z}$ be the basis of the frame A, and let $\boldsymbol{b_x}, \boldsymbol{b_y}, \boldsymbol{b_z}$ be the basis of the frame B. The rotation matrices $\boldsymbol{R}_{B}^{A}$ and $\boldsymbol{R}_{A}^{B}$ can then be expressed by

$$\boldsymbol{R}_{B}^{A} = \left[b_x^A \, b_y^A \, b_z^A \right], \tag{3.2}$$

$$\boldsymbol{R}_{A}^{B} = \left[a_x^B \, a_y^B \, a_z^B \right]. \tag{3.3}$$

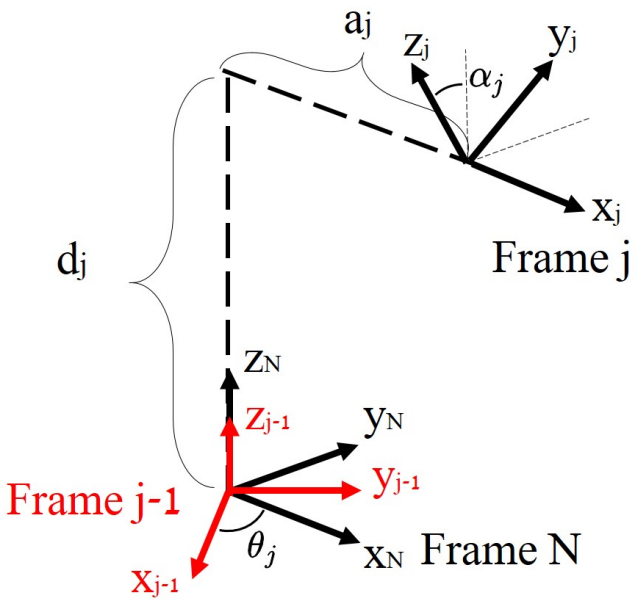


Figure 3.4: Intermediate frame is defined to construct the homogeneous transforamtion with D-H parameter convension

To construct the rotation matrix between fram j-1 and j with the D-H parameters, we define a intermediate frame N as shown in Figure 3.4. The rotation matrix $\boldsymbol{R}_{j-1}^{N}$ is used to describe the orientation change that is caused by the joint angle rotation θ_j. In addition, the rotation matrix $\boldsymbol{R}_{N}^{j}$ represents the twist angle α_j. The rotation matrix $\boldsymbol{R}_{j-1}^{N}$ and $\boldsymbol{R}_{N}^{j}$ are

$$\boldsymbol{R}_{j-1}^{N} = \begin{bmatrix} \cos\theta_j & \sin\theta_j & 0 \\ -\sin\theta_j & \cos\theta_j & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.4}$$

$$\boldsymbol{R}_N^j = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha_j & \sin\alpha_j \\ 0 & -\sin\alpha_j & \cos\alpha_j \end{bmatrix}. \tag{3.5}$$

To obtain $\boldsymbol{R}_j^{j-1}$, we multiply these two rotation matrices and take the transpose operation

$$\boldsymbol{R}_j^{j-1} = \left(\boldsymbol{R}_{j-1}^N \boldsymbol{R}_N^j\right)^\intercal = \begin{bmatrix} \cos\theta_j & -\sin\theta_j\cos\alpha_j & \sin\theta_j\sin\alpha_j \\ \sin\theta_j & \cos\theta_j\cos\alpha_j & -\cos\theta_j\sin\alpha_j \\ 0 & \sin\alpha_j & \cos\alpha_j \end{bmatrix}. \tag{3.6}$$

After the rotation matrix containing the D-H parameters has been derived, it remains to write out the expression for the displacement vector. Since the vector is invariant with respect to coordinate systems, the vector $\boldsymbol{d}_{j-1,j}$ which connects the origin of j-1 frame to the origin of the j frame can be written as the sum direction of the two vectors $\boldsymbol{x}_j$ and $\boldsymbol{z}_{j-1}$ in

$$\boldsymbol{d}_{j-1,j} = a_j\boldsymbol{x}_j + d_j\boldsymbol{z}_{j-1}. \tag{3.7}$$

Now we choose to express the vector $\boldsymbol{d}_{j-1,j}$ in terms of components relative to frame j-1. It is denoted as $\boldsymbol{d}_{j-1,j}^{j-1}$ where

$$\boldsymbol{d}_{j-1,j}^{j-1} = a_j \begin{bmatrix} \cos\theta_j \\ \sin\theta_j \\ 0 \end{bmatrix} + d_j \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} a_j\cos\theta_j \\ a_j\sin\theta_j \\ d_j \end{bmatrix}. \tag{3.8}$$

According to (3.1) , the complete homogeneous transformation between robot body frame j-1 and body frame j can be written as

$$\boldsymbol{H}_j^{j-1} = \begin{bmatrix} \cos\theta_j & -\sin\theta_j\cos\alpha_j & \sin\theta_j\sin\alpha_j & a_j\cos\theta_j \\ \sin\theta_j & \cos\theta_j\cos\alpha_j & -\cos\theta_j\sin\alpha_j & a_j\sin\theta_j \\ 0 & \sin\alpha_j & \cos\alpha_j & d_j \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.9}$$

By (3.9) , the homogeneous transformation between the inertial frame I assigned to the third frame and the camera frame C is written as

$$\boldsymbol{H}_C^I = \boldsymbol{H}_3^I\boldsymbol{H}_4^3\boldsymbol{H}_5^4\boldsymbol{H}_C^5. \tag{3.10}$$

In this section, we have gone through the details of how to construct the homogeneous transformation in keeping with the D-H convention. In the next section , we will explore how the velocity of a selected point on the robot manipulator be calculated.

3.1.3 Linear and angular velocity in the D-H convention

In the framework of image based visual servo control, the differential relationship between the image features in the image plane, and the generalized coordinates need to be derived. The robot generalized coordinates should be able to describe the robot configuration, so the joint angle vector $\boldsymbol{q}_C = \begin{bmatrix} q_3 & q_4 & q_5 \end{bmatrix}^\mathsf{T}$ has been chosen as the generalized coordinates. Because we drive the last three joints of the robot instead of all six joints to maneuver the camera-projector unit, there are only three generalized coordinates. To study the mapping from velocity to image feature velocity in the image plane, we break the mapping into two parts. We first calculate the mapping from derivatives of generalized coordinates to the inertial velocity. We subsequently determine the mapping from inertial velocity to image space velocity. In this section we mainly focus on the first step, and the second step will be explained in detail in the next section.

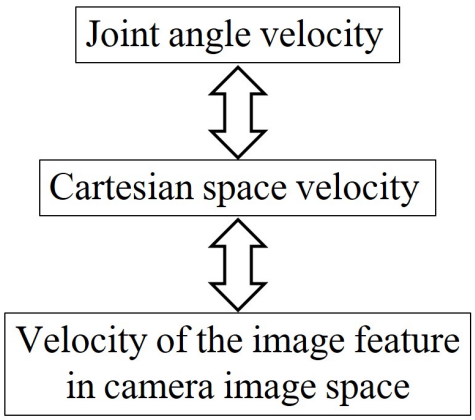


Figure 3.5: The relationship between the mapping

To derive the linear and angular velocity of a selected point on the robot manipulator when all joints are revolute, we start by assuming a selected point p is fixed on a rigid body with frame B , and the rigid body frame B moves in the robot based frame $(\hat{\boldsymbol{x}}_0, \hat{\boldsymbol{y}}_0, \hat{\boldsymbol{z}}_0)$.

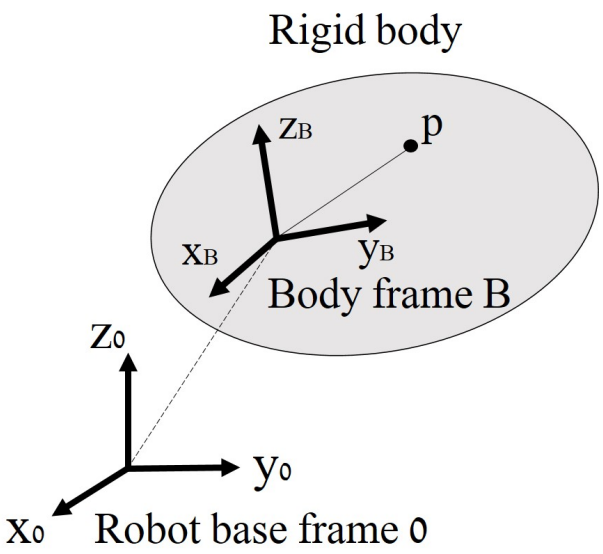


Figure 3.6: Velocity of an arbitrary point on a rigid body

Without choosing any specific coordinates yet, the linear velocity vector of the selected point p on the rigid body B is given by

$$\boldsymbol{v}_{0,p} = \boldsymbol{v}_{0,B} + \boldsymbol{\omega}_{0,B} \times \boldsymbol{d}_{B,p}. \tag{3.11}$$

Then, we apply (3.11) to the case where the body frame B now is link number j-1. Point p is now the origin of link number j. We have

$$\begin{aligned} \boldsymbol{v}_{0,j} &= \boldsymbol{v}_{0,j-1} + \boldsymbol{\omega}_{0,j-1} \times \boldsymbol{d}_{j-1,j}, \\ &= \boldsymbol{v}_{0,j-1} + \boldsymbol{\omega}_{0,j-1} \times (\boldsymbol{r}_{0,j} - \boldsymbol{r}_{0,j-1}). \end{aligned} \tag{3.12}$$

For the angular velocity, **the joint number j drives frame j**. The angular velocity vector of the body frame j with respect to the robot base frame 0 can be computed as

$$\begin{aligned} \boldsymbol{\omega}_{0,j} &= \boldsymbol{\omega}_{0,1} + \boldsymbol{\omega}_{0,2} + ...\boldsymbol{\omega}_{j-2,j-1} + \boldsymbol{\omega}_{j-1,j}, \\ &= \dot{\theta}_1 \hat{\boldsymbol{z}}_0 + \dot{\theta}_2 \hat{\boldsymbol{z}}_1 + ... + \dot{\theta}_{j-1} \hat{\boldsymbol{z}}_{j-2} + \dot{\theta}_j \hat{\boldsymbol{z}}_{j-1}, \end{aligned} \tag{3.13}$$

where $\boldsymbol{\omega}_{j-1,j}$ is angular velocity of frame j with respect to the frame $j-1$.

In this section the linear and angular velocity vector of an arbitrary point on the robot manipulator has been derived in Equations (3.12) and (3.13). In addition, the linear and

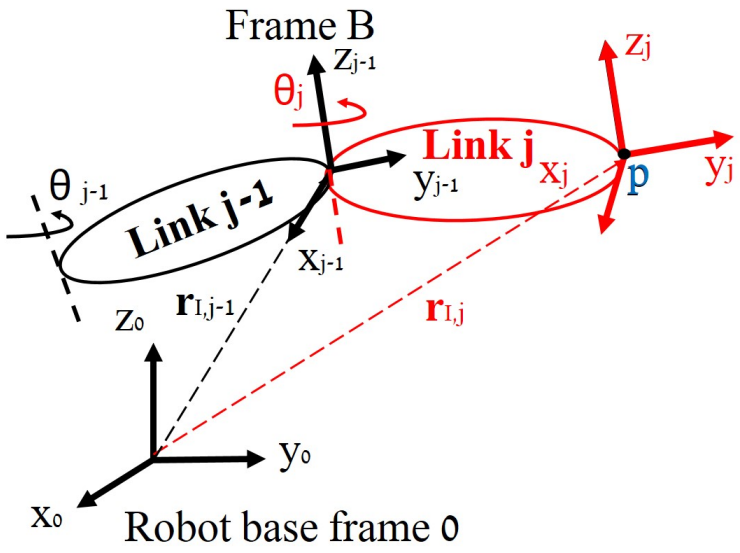


Figure 3.7: Velocity of an arbitrary point on a rigid body

angular velocity vectors of the current body frame are expressed in terms of the velocity vectors of the previous body frame. This recursive expression is advantageous in deriving the kinematic Jacobian matrix for the robot manipulator in the next section.

3.1.4 Manipulator kinematic Jacobian

In this section, we will derive the robot manipulator kinematic Jacobian based on the Equations (3.12) and (3.13). The kinematic Jacobian of the robot manipulator is a differential mapping that maps from the derivatives of the generalized coordinates to the inertial velocity of point p.

To find the end effector linear and angular velocity contribution from only the joint number j, we fix all other joints except the joint number j. After fixing the other joints, the robot manipulator now is seperated into two rigid body parts as shown in Figure 3.8: The first rigid body part includes joint numbers j=1...j-1, which are assumed to be fixed and can not not rotates. The second part includes joint numbers j=j...N, which are fixed together on the same rigid body and likewise can not rotate relative to one another. According to this constraint, the $\boldsymbol{\omega}_{j-1}$ term in (3.13) vanishes. Expressed relative to in the robot base frame 0, the angular velocity of the robot end effector body frame N is

$$\boldsymbol{\omega}_N^0 = \boldsymbol{\omega}_j^0 = \dot{\theta}_j \hat{\boldsymbol{z}}_{j-1}^0. \tag{3.14}$$

Under the same assumptions, the velocity term $\boldsymbol{v}_{0,j-1}$ in Equation (3.12) also vanishes be-

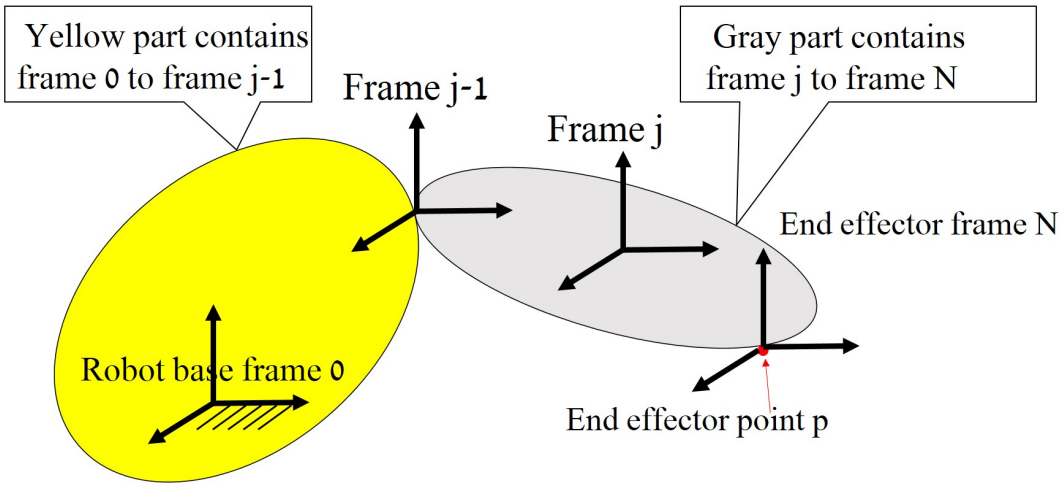


Figure 3.8: Robot manipulator is seperated into two rigid bodies: yellow part and grey part. The yellow part fix on the ground, so only the grey part can move.

cause frame $j-1$ is rigidly connected to the ground. Express in the base coordinate frame 0, the linear velocity vector that is contributed by the change in joint j can now be written as

$$\begin{aligned}\boldsymbol{v}_{0,p}^{0} &= \cancelto{0}{\boldsymbol{v}_{0,j-1}^{0}} + \boldsymbol{\omega}_{0,p}^{0} \times \left(\boldsymbol{r}_{0,j}^{0} - \boldsymbol{r}_{0,j-1}^{0}\right) \\ &= \boldsymbol{\omega}_{0,j}^{0} \times \left(\boldsymbol{r}_{0,j}^{0} - \boldsymbol{r}_{0,j-1}^{0}\right) \\ &= \boldsymbol{\omega}_{j-1,j}^{0} \times \left(\boldsymbol{r}_{0,j}^{0} - \boldsymbol{r}_{0,j-1}^{0}\right) \\ &= \dot{\theta}_j \hat{\boldsymbol{z}}_{j-1}^{0} \times \left(\boldsymbol{r}_{0,j}^{0} - \boldsymbol{r}_{0,j-1}^{0}\right). \end{aligned} \tag{3.15}$$

Note that in Equation (3.15), $\boldsymbol{\omega}_{0,p}^{0} = \boldsymbol{\omega}_{0,j}^{0} = \boldsymbol{\omega}_{j-1,j}^{0}$ because frame 0 and frame j-1 are both by assumption rigidly connected to the ground. To write these equations into the form of a Jacobian matrix, we rearrange (3.14) and (3.15) as

$$\boldsymbol{\omega}_{N}^{0} = \hat{\boldsymbol{z}}_{j-1}^{0} \dot{\theta}_j = \mathscr{J}_{\omega,j} \dot{\boldsymbol{q}}_j, \tag{3.16}$$

$$\boldsymbol{v}_{0,p}^{0} = \left\{ \hat{\boldsymbol{z}}_{j-1}^{0} \times \left(\boldsymbol{r}_{0,j}^{0} - \boldsymbol{r}_{0,j-1}^{0}\right) \right\} \dot{\theta}_j = \mathscr{J}_{v,j} \dot{\boldsymbol{q}}_j. \tag{3.17}$$

Note that $\mathscr{J}_{\omega,j} = \hat{\boldsymbol{z}}_{j-1}^{0}$ and $\mathscr{J}_{v,j} = \left\{ \hat{\boldsymbol{z}}_{j-1}^{0} \times \left(\boldsymbol{r}_{0,j}^{0} - \boldsymbol{r}_{0,j-1}^{0}\right) \right\}$ are both 3×1 vectors. Now stacking (3.17) and (3.16) into a single matrix, we have

$$\begin{aligned}\begin{bmatrix}\boldsymbol{v}_{0,p}^{0}\\ \boldsymbol{\omega}_{0,p}^{0}\end{bmatrix}_{6\times 1} &= \begin{bmatrix}\mathscr{J}_{v,1} & \mathscr{J}_{v,2} & \cdots & \mathscr{J}_{v,N}\\ \mathscr{J}_{\omega,1} & \mathscr{J}_{\omega,2} & \cdots & \mathscr{J}_{\omega,N}\end{bmatrix}_{6\times N}\begin{bmatrix}\dot{q}_1\\ \dot{q}_2\\ \vdots\\ \dot{q}_N\end{bmatrix}\\ &= \begin{bmatrix}\hat{\boldsymbol{z}}_0^0\times\left(\boldsymbol{r}_{0,1}^0-\boldsymbol{r}_{0,0}^0\right) & \cdots & \hat{\boldsymbol{z}}_{j-1}^0\times\left(\boldsymbol{r}_{0,j}^0-\boldsymbol{r}_{0,j-1}^0\right)\\ \hat{\boldsymbol{z}}_0^0 & \cdots & \hat{\boldsymbol{z}}_{j-1}^0\end{bmatrix}_{6\times N}\begin{bmatrix}\dot{q}_1\\ \vdots\\ \dot{q}_N\end{bmatrix}\\ &= \boldsymbol{J}\dot{\boldsymbol{q}}. \end{aligned} \tag{3.18}$$

In (3.18) , the components $\hat{\boldsymbol{z}}_{j-1}^0$ represent the z axis of frame $j-1$ in terms of the basis of 0. These can be obtained by the third column of the rotation matrix multiplication, given as

$$\boldsymbol{z}_{j-1}^0 = \boldsymbol{R}_1^0\left(q_1\right)...\boldsymbol{R}_{j-1}^{j-2}\begin{bmatrix}0\\0\\1\end{bmatrix}. \tag{3.19}$$

In addition, the displacement vector $\boldsymbol{r}_{0,j}^0$ can be found as the first three elements in the fourth column of the homogeneous transformation, and it is given as

$$\begin{bmatrix}\boldsymbol{r}_{0,j}^0\\1\end{bmatrix} = \boldsymbol{H}_1^0\boldsymbol{H}_2^1...\boldsymbol{H}_j^{j-1}\begin{bmatrix}0\\0\\0\\1\end{bmatrix}. \tag{3.20}$$

From (3.19) and (3.20) , all element in the Jacobian matrix in Equation (3.18) can be found. Finally, the matrix $\boldsymbol{J}$ in Equation (3.18) is known as the kinematic Jacobian of the robot manipulator.

3.1.5 ABB IRB120 robot arm kinematics

In the previous sections, a method for describing the robot kinematics has been introduced. In our study, the camera and projector together form the robot end effector, and the robot end effector frame coincides with the camera. For the case when the camera-projector unit is perfectly mounted by the user, this perfectly mounted camera frame is defined as $\widetilde{C}$. The D-H parameters of the ideally mounted robot end effector are listed in the table below:

Table 3.2: Denavit-Hartenberg parameters of the ideally mounted camera-projector unit

Link	$\theta_j(rad)$	$\alpha_j(rad)$	$a_j(mm)$	$d_j(mm)$
6	θ_6	$\alpha = 0.01$	$d = 200$	$\delta = 0$

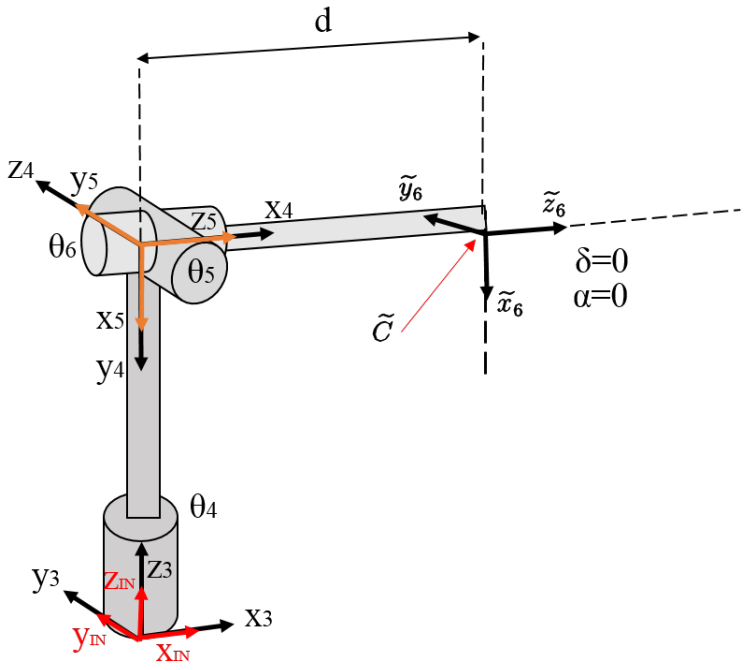


Figure 3.9: Perfect mounting position $\widetilde{C}$

Since we only drive the last three joints of the robot arm, as we described in previous sections, the inertial frame I is identical to frame number 3. The homogeneous transformation between the inertial frame I and the robot body frame C_3 is consequently

$$\boldsymbol{H}^I_{C_3} = \begin{bmatrix} \boldsymbol{R}^I_{C_3} & \boldsymbol{r}^I_{C_3} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.21}$$

Now combine Equation (3.21) and Equation (3.18) . We obtain the Jacobian matrix that maps the derivatives of the generalized coordinates into the camera frame inertial velocity as

$$\begin{aligned}\begin{bmatrix}\boldsymbol{v}^I_{I,\widetilde{C}}\\ \boldsymbol{\omega}^I_{I,\widetilde{C}}\end{bmatrix} &= \begin{bmatrix}\hat{\boldsymbol{z}}^I_{C_3}\times\left(\boldsymbol{r}^I_{I,\widetilde{C}}-\boldsymbol{r}^I_{I,C_3}\right) & \hat{\boldsymbol{z}}^I_{C_4}\times\left(\boldsymbol{r}^I_{I,\widetilde{C}}-\boldsymbol{r}^I_{I,C_4}\right) & \hat{\boldsymbol{z}}^I_{C_5}\times\left(\boldsymbol{r}^I_{I,\widetilde{C}}-\boldsymbol{r}^I_{I,C_5}\right)\\ \hat{\boldsymbol{z}}^I_{C_3} & \hat{\boldsymbol{z}}^I_{C_4} & \hat{\boldsymbol{z}}^I_{C_5}\end{bmatrix}\begin{bmatrix}\dot{q}_4\\ \dot{q}_5\\ \dot{q}_6\end{bmatrix},\\ &= \boldsymbol{J}_q\dot{\boldsymbol{q}}_{\widetilde{C}}.\end{aligned} \tag{3.22}$$

The symbols $\hat{\boldsymbol{z}}^I_{C_3}, \hat{\boldsymbol{z}}^I_{C_4}, \hat{\boldsymbol{z}}^I_{C_5}$ are the representations in terms of the inertial basis of the $\hat{\boldsymbol{z}}$ axis of frames C_3, C_4, C_5, respectively. They can be calculated in terms of rotation matrices using Equation (3.19) as

$$\begin{aligned}\hat{\boldsymbol{z}}^I_{C_3} &= \boldsymbol{R}^I_{C_3}\begin{bmatrix}0 & 0 & 1\end{bmatrix}^T,\\ \hat{\boldsymbol{z}}^I_{C_4} &= \boldsymbol{R}^I_{C_3}\boldsymbol{R}^{C_3}_{C_4}(q_4)\begin{bmatrix}0 & 0 & 1\end{bmatrix}^T, \text{ and}\\ \hat{\boldsymbol{z}}^I_{C_5} &= \boldsymbol{R}^I_{C_3}\boldsymbol{R}^{C_3}_{C_4}(q_4)\,\boldsymbol{R}^{C_4}_{C_5}(q_5)\begin{bmatrix}0 & 0 & 1\end{bmatrix}^T.\end{aligned} \tag{3.23}$$

In addition, $\boldsymbol{r}^I_{I,C_3}, \boldsymbol{r}^I_{I,C_4}, \boldsymbol{r}^I_{I,C_5}, \boldsymbol{r}^I_{I,C_6}$ are representations of the vectors from the origin of frame I to the origin of frames C_3, C_4, C_5, C_6, respectively. They can be obtained by the multiplication of the homogeneous transformation matrices in the expressions

$$\begin{aligned}\begin{bmatrix}\boldsymbol{r}^I_{I,C_3}\\ 1\end{bmatrix} &= \boldsymbol{H}^I_{C_3}\begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix}^T,\\ \begin{bmatrix}\boldsymbol{r}^I_{I,C_4}\\ 1\end{bmatrix} &= \boldsymbol{H}^I_{C_3}\boldsymbol{H}^{C_3}_{C_4}(q_4)\begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix}^T,\\ \begin{bmatrix}\boldsymbol{r}^I_{I,C_5}\\ 1\end{bmatrix} &= \boldsymbol{H}^I_{C_3}\boldsymbol{H}^{C_3}_{C_4}(q_4)\,\boldsymbol{H}^{C_4}_{C_5}(q_5)\begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix}^T, \text{ and}\\ \begin{bmatrix}\boldsymbol{r}^I_{I,\widetilde{C}}\\ 1\end{bmatrix} &= \boldsymbol{H}^I_{C_3}\boldsymbol{H}^{C_3}_{C_4}(q_4)\,\boldsymbol{H}^{C_4}_{C_5}(q_5)\,\boldsymbol{H}^{C_5}_{\widetilde{C}}(q_6)\begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix}^T.\end{aligned} \tag{3.24}$$

The robot manipulator kinematics has been introduced in this section. In general, the kinematics of the robot arm can be systematically derived once the Denavit–Hartenberg parameters are assigned. In the next section, we will explore the pinhole camera model, which is the model chosen to represent the camera in our study.

3.1.6 Pinhole camera model

In our study, the pinhole camera model with prespective projection is presented. The prespective projection model is depicted schematically in Figure 3.10 where the image of the object is in front of the camera. This arrangement is referred to as the front projection

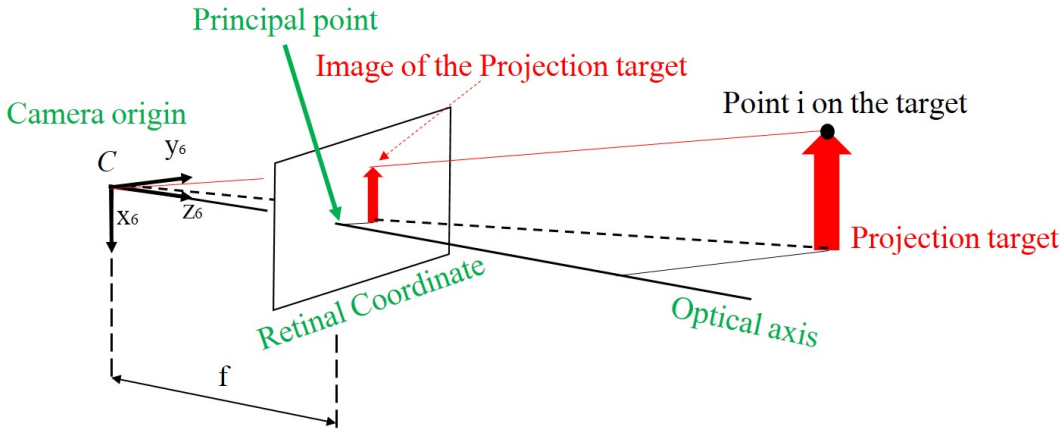


Figure 3.10: Camera model with perspective projection

pinhole model.The components of the position vector of a point i on the target T relative to the camera frame C is $\boldsymbol{r}_{C,i}^{C} = \left[X_{C,i} Y_{C,i} Z_{C,i}\right]^{\intercal}$. Therefore, the image of point i in the camera retinal coordinate with a camera having focal length f can be expressed as

$$\begin{aligned} u_i &= f\frac{X_{C,i}}{Z_{C,i}}, \\ v_i &= f\frac{Y_{C,i}}{Z_{C,i}}. \end{aligned} \tag{3.25}$$

However, the map from Cartesian space to camera retinal coordinates must be further modified because the image of the real camera is formed in pixel coordinates, and the image in the pixel coordinates is scaled and distorded with respect to the retinal coordinates. Figure 3.11 shows how the pixel coordinates are defined. We use the following parameters to describe the pixel coordinates: the pixel space scaling factors s_u, s_v ,the shearing of the pixel s_θ. The shearing of the pixel s_θ describes how the pixel is sheared relative to the vertical and horizontal direction. The location of the principal point or the image of the line of sight of the camera in the CCD array, o_u, o_v. Along with these parameters, the mapping of a single point i from the retinal coordinates (u_i, v_i) to the pixel coordinates (ξ_i, η_i) can be written as

$$\begin{aligned} \xi_i &= f s_u u_i + f s_\theta v_i + o_u, \\ \eta_i &= f s_v v_i + o_v. \end{aligned} \tag{3.26}$$

If we write Equation (3.26) as a homogeneous transformation, and apply Equation (3.25), we can map a point with its coordinates $(X_{C,i}, Y_{C,i}, Z_{C,i})$ into the pixel coordinate (ξ_i, η_i) as

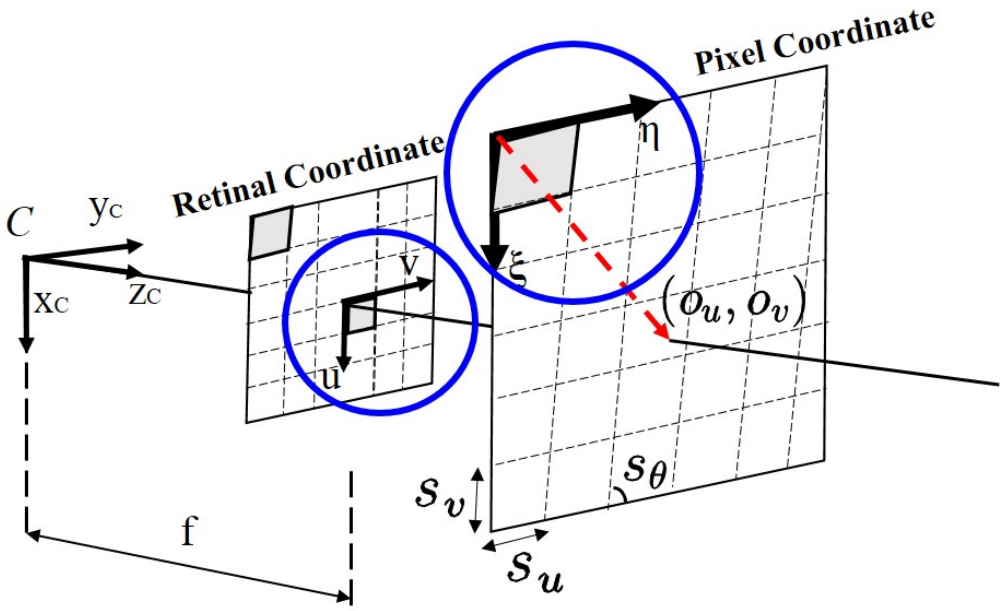


Figure 3.11: Camera retinal coordinate and pixel coordinate

$$Z_{C,i}\begin{bmatrix}\xi_i\\ \eta_i\\ 1\end{bmatrix} = \begin{bmatrix}fs_u & fs_v & o_u\\ 0 & fs_v & o_v\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}X_{C,i}\\ Y_{C,i}\\ Z_{C,i}\\ 1\end{bmatrix}. \tag{3.27}$$

3.1.7 Image feature interaction matrix

The interaction matrix of the image feature is crutial for constructing the image Jacobian matrix in the next section. The feature of the image can be various type, they can be lines , points, or centre of the image. Thus, the purpose of finding the interaction matrix for the certain image feature is to relate the image space velocity of the feature with the Cartesian space velocity of the camera.

For the **camera off hand with image feature is a point** , projection target point i is rigidly fixed on the robot end effector, but the camera is fixed on the inertial frame I *(so the camera is off hand)*. In this scenario, the velocity and angular velocity of the end effector e relative to the camera frame C are $\boldsymbol{v}^C_{C,e}$ and $\boldsymbol{\omega}^C_{C,i}$ respectively. In addition, the coordinate of point i position the vector $\boldsymbol{r}^C_{C,i}$ in camera frame is $\begin{bmatrix}X_{C,i} & Y_{C,i} & Z_{C,i}\end{bmatrix}^\intercal$, then we will have the relationship

$$\dot{\boldsymbol{r}}_{C,i}^{C} = \boldsymbol{\omega}_{C,e}^{C} \times \boldsymbol{r}_{C,i}^{C} + \boldsymbol{v}_{C,e}^{C} \tag{3.28}$$

Furthermore, for the **camera in hand problem when image feature is a point**, the form of the point velocity remains the same as in Equation (3.28) but there is a sign change. The reason for the change in the sign is that the projection target always moves in the opposite direction in the image plane when the camera is fixed on a mobile end effector. For this reason, Equation (3.28) becomes

$$\dot{\boldsymbol{r}}_{C,i}^{C} = -\boldsymbol{\omega}_{I,C}^{C} \times \boldsymbol{r}_{C,i}^{C} - \boldsymbol{v}_{I,C}^{C}. \tag{3.29}$$

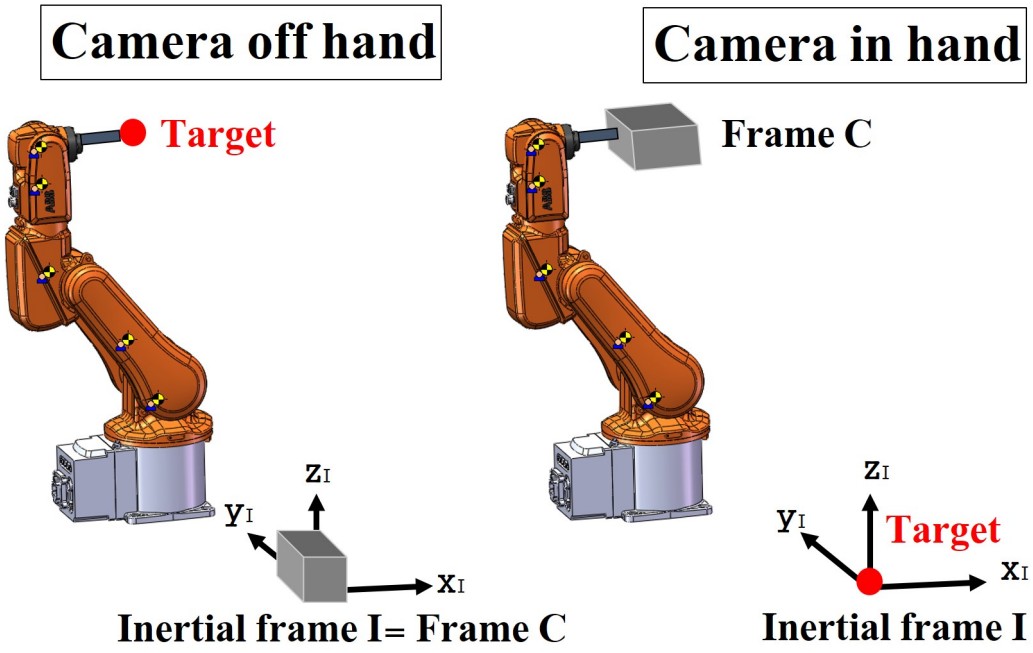


Figure 3.12: Camera in hand vs camera off hand

Write the Equation (3.29) in scalar form, we have

$$\begin{aligned}
\dot{X}_{C,i} &= Y_{C,i}\omega_{z_{I,C}}^{C} - Z_{C,i}\omega_{y_{I,C}}^{C} - v_{x_{I,C}}^{C} \\
\dot{Y}_{C,i} &= Z_{C,i}\omega_{x_{I,C}}^{C} - X_{C,i}\omega_{z_{I,C}}^{C} - v_{y_{I,C}}^{C} \\
\dot{Z}_{C,i} &= X_{C,i}\omega_{y_{I,C}}^{C} - Y_{C,i}\omega_{x_{I,C}}^{C} - v_{z_{I,C}}^{C}.
\end{aligned} \tag{3.30}$$

Furthermore, the derivative of (3.25) is expressed in the form

$$\begin{aligned}
\begin{bmatrix}\dot{u}_i\\ \dot{v}_i\end{bmatrix} &= \frac{f}{Z_{C,i}}\begin{bmatrix}\dot{X}_{C,i}\\ \dot{Y}_{C,i}\end{bmatrix} - \frac{f}{Z_{C,i}^2}\dot{Z}_{C,i}\begin{bmatrix}X_{C,i}\\ Y_{C,i}\end{bmatrix},\\
&= \frac{f}{Z_{C,i}}\begin{bmatrix}\dot{X}_{C,i}\\ \dot{Y}_{C,i}\end{bmatrix} - \frac{f\dot{Z}_{C,i}}{Z_{C,i}}\left(\frac{1}{Z_{C,i}}\begin{bmatrix}X_{C,i}\\ Y_{C,i}\end{bmatrix}\right),\\
&= \frac{f}{Z_{C,i}}\begin{bmatrix}\dot{X}_{C,i}\\ \dot{Y}_{C,i}\end{bmatrix} - \frac{f\dot{Z}_{CC,p}}{Z_{C,i}}\begin{bmatrix}u_{C,i}\\ v_{C,i}\end{bmatrix}, \text{ or}\\
&= \frac{f}{Z_{C,i}}\begin{bmatrix}1 & 0 & -u_{C,i}\\ 0 & 1 & -v_{C,i}\end{bmatrix}\begin{bmatrix}\dot{X}_{C,i}\\ \dot{Y}_{C,i}\\ \dot{Z}_{C,i}\end{bmatrix}.
\end{aligned} \tag{3.31}$$

Now substitute Equation (3.30) into Equation (3.31) we obtain the interaction matrix for a feature point in the following expression

$$\begin{bmatrix}\dot{u}_i\\ \dot{v}_i\end{bmatrix} = \frac{f}{Z_{C,i}}\begin{bmatrix}1 & 0 & -u_{C,i}\\ 0 & 1 & -v_{C,i}\end{bmatrix}\begin{bmatrix}-1 & 0 & 0 & 0 & -Z_{C,i} & Y_{C,i}\\ 0 & -1 & 0 & Z_{C,i} & 0 & -X_{C,i}\\ 0 & 0 & -1 & -Y_{C,i} & X_{C,i} & 0\end{bmatrix}\begin{bmatrix}\boldsymbol{v}_{I,C}^{C}\\ \boldsymbol{\omega}_{I,C}^{C}\end{bmatrix}. \tag{3.32}$$

3.1.8 Image Jacobian for point feature without pixel shearing effect

In the previous section, the camera model with pixel shearing effect has been derived. In this section, we focus on constructing the image Jacobian mapping from the inertial velocity to image space velocity without considering the pixel shearing effect. From the previous section, we have already obtain the time derivative of Equation (3.25), which is given by Equation 3.31. In addition, from Equation (3.26), the transformation without considering the shearing effect is

$$\begin{aligned}
\zeta &= s_u u + o_u,\\
\eta &= s_v v + o_v.
\end{aligned} \tag{3.33}$$

We take time derivative of Equation (3.33) and find that

$$\begin{aligned}
\dot{\zeta} &= s_u \dot{u},\\
\dot{\eta} &= s_v \dot{v}.
\end{aligned} \tag{3.34}$$

In addition, when we combine (3.32) and (3.34), we have

$$\begin{bmatrix}\dot{\zeta}_i\\ \dot{\eta}_i\end{bmatrix} = \underbrace{\frac{f}{Z_{C,i}}\begin{bmatrix}s_u & 0\\ 0 & s_v\end{bmatrix}\begin{bmatrix}1 & 0 & -u_{C,i}\\ 0 & 1 & -v_{C,i}\end{bmatrix}\begin{bmatrix}-1 & 0 & 0 & 0 & -Z_{C,i} & Y_{C,i}\\ 0 & -1 & 0 & Z_{C,i} & 0 & -X_{C,i}\\ 0 & 0 & -1 & -Y_{C,i} & X_{C,i} & 0\end{bmatrix}}_{Image\ Jacobian\ matrix}\begin{bmatrix}\boldsymbol{v}_{I,C}^{C}\\ \boldsymbol{\omega}_{I,C}^{C}\end{bmatrix}$$

For the case where two points on the image are selected as the image feature, the form of the corresponding image Jacobian is obtained by simply stacking Equation (3.35) twice. It then follows that

$$\begin{aligned}\dot{\boldsymbol{s}} = \begin{bmatrix}\dot{\zeta}_1\\ \dot{\eta}_1\\ \dot{\zeta}_2\\ \dot{\eta}_2\end{bmatrix} &= \frac{f}{Z_{\widetilde{C},i}}\begin{bmatrix}s_u & 0 & 0 & 0\\ 0 & s_v & 0 & 0\\ 0 & 0 & s_u & 0\\ 0 & 0 & 0 & s_v\end{bmatrix}\begin{bmatrix}\frac{-1}{Z_{\widetilde{C},1}} & 0 & \frac{u_1}{Z_{\widetilde{C},1}} & u_1v_1 & -(1+u_1^2) & v_1\\ 0 & \frac{-1}{Z_{\widetilde{C},1}} & \frac{v_1}{Z_{\widetilde{C},1}} & (1+v_1^2) & -u_1v_1 & -u_1\\ \frac{-1}{Z_{\widetilde{C},2}} & 0 & \frac{u_2}{Z_{\widetilde{C},2}} & u_2v_2 & -(1+u_2^2) & v_2\\ 0 & \frac{-1}{Z_{\widetilde{C},2}} & \frac{v_2}{Z_{\widetilde{C},2}} & (1+v_2^2) & -u_2v_2 & -u_2\end{bmatrix}\begin{bmatrix}\boldsymbol{v}_{I,\widetilde{C}}^{\widetilde{C}}\\ \boldsymbol{\omega}_{I,\widetilde{C}}^{\widetilde{C}}\end{bmatrix}\\ &= \boldsymbol{J}_s\begin{bmatrix}\boldsymbol{v}_{I,\widetilde{C}}^{\widetilde{C}}\\ \boldsymbol{\omega}_{I,\widetilde{C}}^{\widetilde{C}}\end{bmatrix}.\end{aligned} \tag{3.35}$$

In Equation (3.35), the terms $Z_{\widetilde{C},1}$ and $Z_{\widetilde{C},1}$ are the range between the camera and two target points. In the thesis, we obtain these two range value from the feedback of the GPS sensor.

3.1.9 Total kinematic Jacobian

In the above sections, we have already derived the robot manipulator kinematic Jacobian (3.22) and image feature Jacobian (3.35). In Equation (3.22) , the inertial velocity of the end effector is expressed in the inerital frame I, however, the end effector velocity of (3.35) is expressed in the components relative to the camera frame $\widetilde{C}$. After all, in order to build the complete mapping from the derivatives $\dot{\boldsymbol{q}}_{\widetilde{C}}$ to the image feature velocity $\dot{\boldsymbol{s}}$, it is necessary to perform a change of the coordinates.

For an arbitrary vector $\boldsymbol{p}$ in Cartesian space, the components of vector $\boldsymbol{p}$ in the basis of frame A is $\boldsymbol{p}^A$. In the same way, we can express the same vector in the frame B which is denoted as $\boldsymbol{p}^B$. In addition, the coordinate transformation from the frame B to the frame A is given by

$$\boldsymbol{p}^A = \boldsymbol{R}_B^A\boldsymbol{p}^B. \tag{3.36}$$

where $\boldsymbol{R}_B^A$ is the rotation matrix from frame B to frame A. From the coordinate transformation property, Equation (3.22) can be expressed in the ideally mounted camera frame $\widetilde{C}$ as

$$\begin{aligned}\begin{bmatrix}\boldsymbol{v}_{I,\widetilde{C}}^{\widetilde{C}}\\ \boldsymbol{\omega}_{I,\widetilde{C}}^{\widetilde{C}}\end{bmatrix} &= \begin{bmatrix}\boldsymbol{R}_I^{\widetilde{C}} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{R}_I^{\widetilde{C}}\end{bmatrix}\begin{bmatrix}\boldsymbol{v}_{I,\widetilde{C}}^{I}\\ \boldsymbol{\omega}_{I,\widetilde{C}}^{I}\end{bmatrix}\\ &= \begin{bmatrix}\boldsymbol{R}_I^{\widetilde{C}} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{R}_I^{\widetilde{C}}\end{bmatrix}\boldsymbol{J}_q\dot{\boldsymbol{q}}_{\widetilde{C}}\end{aligned} \tag{3.37}$$

Comebine Equation (3.35) with Equation (3.37), the complete Jacobian maps from the derivatives of the generalized coordinates to the image velocity in the image plane is

$$\dot{\boldsymbol{s}} = \boldsymbol{J}_s\begin{bmatrix}\boldsymbol{R}_I^{\widetilde{C}} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{R}_I^{\widetilde{C}}\end{bmatrix}\boldsymbol{J}_q\dot{\boldsymbol{q}}_{\widetilde{C}}. \tag{3.38}$$

Now we have finished the derivation of the robotic system kinematics that includes the robot manipulator and the camera-projector unit. In the next section, we will mainly focus on deriving a dynamic model for the robotic system.

3.2 Robot dynamics

Dynamic modeling plays an essential part in constructing a simulation that enables us to test the control strategy without resulting to experiment. In this chapter, we use a Lagrangian formulation to obtain the dynamic model of the robot system in joint space. Initially, we will focus on the Largrangian formulation of motion of the robot manipulator systematically. In the second part of the section, we will identify those dynamic parameters of the `ABB IRB120` robot from the CAD model that is available on `ABB` company website.

3.2.1 Largrange's equations for unconstrained systems

For a system of particles subject to conservative forces described by a potential, when we choose the inertial Cartesian coordinate space as the general coordinates, Newton's second law states thatfor the i^{th} particale we have

$$m\ddot{x}_i = F_i. \tag{3.39}$$

The right hand side of the equation is the conservative force represented by the potential energy. It can be written as $\frac{-\partial \mathscr{V}}{\partial x_i}$. we define the Lagrangian $\mathscr{L}$ to be given by $\mathscr{L} = \mathscr{T} - \mathscr{V}$. It can be shown that the Equation (3.39) can be used to derive Lagrange's equations for a conservative system

$$\frac{d}{dt}\frac{\partial \mathscr{L}}{\partial \dot{x}_i} - \frac{\partial \mathscr{L}}{\partial x_i} = 0 \qquad (i = 1, 2, ...n)\,. \tag{3.40}$$

when we perform the change of coordinates from the inertial to joint space coordinates, the inertial Cartesian coordinates and joint space coordinates aer assumed to satisfy the equations

$$x_i = x_i\left(q_1, ..., q_N, t\right), \tag{3.41}$$
$$q_j = q_j\left(x_1, ..., x_n, t\right). \tag{3.42}$$

Since the form of Lagrange's equations is invariant with respect to change of generalized coordinates, the unconstrained system Lagrangian equations in joint space coordinate can be written as

$$\frac{d}{dt}\frac{\partial \mathscr{L}}{\partial \dot{q}_j} - \frac{\partial \mathscr{L}}{\partial q_j} = 0 \qquad (j = 1, 2, ...N) \tag{3.43}$$

3.2.2 Largrange's equations for holonomic constrained systems

When formulating Lagrangian mechanics we usually consider two types of constraints: holonomic and nonholonomic. In this thesis we only study holonomic constraints. Holonomic constraints are written in the form

$$\mathbf{\Phi}\left(\boldsymbol{r}_1, ...\boldsymbol{r}_N, t\right) = \mathbf{0} \tag{3.44}$$

where the $\boldsymbol{r}_i$ indicates the displacement of a constrained particle i in the Cartesian coordinates. When a constraint exists between particles, there exists a constraint force that acts on the particle i is denoted as F_i^C. We will refer to external forces, ones that are not associated with constraints as F_i^D, and these are the dynamic forces. In addition, from Newton's Law, the time rate of change of the momentum on a particle i is related to the force by the expression

$$\dot{\boldsymbol{p}}_i = F_i = F_i^C + F_i^D = \mathbf{0}. \tag{3.45}$$

We rearrange this equation and multiply both side with as arbitrary virtual displacement $\delta \boldsymbol{r}_i$. We then have

$$\sum_i^n \left(F_i^D - \dot{\boldsymbol{p}}_i\right) \cdot \delta \boldsymbol{r}_i = \underbrace{-\sum_i^n F_i^C \cdot \delta \boldsymbol{r}_i = \mathbf{0}}_{honolomic\ constraint}. \tag{3.46}$$

The equality on the right side of the above equation holds because the honolomic constraints perform no virtual work. Thus we conclude that for a system subjects the holonomic constraints , it holds that

$$\sum_i^n \left(F_i^D - \dot{\boldsymbol{p}}_i\right) \cdot \delta \boldsymbol{r}_i = \mathbf{0}. \tag{3.47}$$

The above equation (3.47) is known as **D'Alembert's Principle**. We do not need to include the constraint forces F_i^C in the computation of the virtual work when the constraints are holonomic. To derive the Lagrange's equations for holonomically constrained systems based on D'Alembert's principle, we assume that $\boldsymbol{r}_i = \boldsymbol{r}_i\left(q_1, q.., q_N, t\right)$ where $i = 1, 2...n$ is the constrained particle index, and $j = 1, 2...N$ is the general coordinate index, and all q_j are independent. Therefore, the number of holonomic constraints is given by $N = 3n - k$. We can separate (3.47) into two parts as

$$\underbrace{\sum_i^n F_i^D \cdot \delta \boldsymbol{r}_i}_{\mathscr{A}} - \underbrace{\sum_i^n \dot{\boldsymbol{p}}_i \cdot \delta \boldsymbol{r}_i}_{\mathscr{B}} = \mathbf{0}. \tag{3.48}$$

To compute the part $\mathscr{A}$, we take the total differential of $\boldsymbol{r}_i$ and compute the change of this vector in one time step. We have

$$\Delta \boldsymbol{r}_i = \sum_j^N \frac{\partial \boldsymbol{r}_i}{\partial q_j} \Delta q_j + \frac{\partial \boldsymbol{r}_i}{\partial t} \Delta t. \tag{3.49}$$

We then divide both sides of the equation by Δt, and we will thereby generate a velocity expression as

$$\boldsymbol{v}_i = \sum_j^N \frac{\partial \boldsymbol{r}_i}{\partial q_j} \dot{q}_j + \frac{\partial \boldsymbol{r}_i}{\partial t}. \tag{3.50}$$

For a virtual displacement that is assumed to take place instantaneously so that $\Delta t = 0$, we obtain

$$\delta \boldsymbol{r}_i = \sum_j^N \frac{\partial \boldsymbol{r}_i}{\partial q_j} \delta q_j. \tag{3.51}$$

Now we apply Equation(3.51) to part $\mathscr{A}$ in (3.48). We then have

$$\sum_i^n F_i^D \delta \boldsymbol{r}_i = \sum_j^N \sum_i^n F_i^D \frac{\partial \boldsymbol{r}_i}{\partial q_j} \delta q_j = \sum_j^N Q_j \cdot \delta q_j. \tag{3.52}$$

The term Q_j represents the dynamic force in the expression above. To calculate part $\mathscr{B}$ in Equation (3.48) , differentiate Equation (3.50) with respect to $\dot{q}_j$ to yield

$$\frac{\partial \boldsymbol{v}_i}{\partial \dot{q}_j} = \frac{\partial \boldsymbol{r}_i}{\partial q_j}, \tag{3.53}$$

and also the differentiate Equation (3.50) with respect to q_j to get

$$\begin{aligned} \frac{\partial \boldsymbol{v}_i}{\partial q_j} &= \sum_j^N \frac{\partial}{\partial q_j} \left(\frac{\boldsymbol{r}_i}{\partial q_j} \right) \dot{q}_j + \frac{\partial}{\partial t} \frac{\partial \boldsymbol{r}_i}{\partial q_j} \\ &= \frac{d}{dt} \frac{\partial \boldsymbol{r}_i}{\partial q_j}. \end{aligned} \tag{3.54}$$

The last equality in Equation (3.54) comes from the total time derivative of the vector $\frac{\partial \boldsymbol{r}_i}{\partial q_j}$. Therefore we apply (3.54) and (3.53), and the part $\mathscr{B}$ in Equation (3.48) can be expressed as

$$\begin{aligned}
\sum_i^n \dot{\boldsymbol{p}}_i \cdot \delta \boldsymbol{r}_i &= \sum_i^n \frac{d\boldsymbol{p}_i}{dt}\frac{\partial \boldsymbol{r}_i}{\partial q_j}\delta q_j, \\
&= \sum_j^N \frac{d}{dt}\left(\sum_i^n \boldsymbol{p}_i \frac{\boldsymbol{r}_i}{\partial q_j}\right)\delta \cdot q_j - \sum_j^N \sum_i^n \boldsymbol{p}_i \left(\frac{d}{dt}\frac{\partial \boldsymbol{r}_i}{\partial q_j}\right)\delta q_j, \\
&= \sum_j^N \frac{d}{dt}\left(\sum_i^n \boldsymbol{p}_i \frac{\boldsymbol{v}_i}{\partial \dot{q}_j}\right)\delta \cdot q_j - \sum_j^N \sum_i^n \boldsymbol{p}_i \left(\frac{\partial \boldsymbol{v}_i}{\partial q_j}\right)\delta q_j, \\
&= \sum_j^N \left\{\frac{d}{dt}\sum_i^n m_i \boldsymbol{v}_i \cdot \frac{\boldsymbol{v}_i}{\partial \dot{q}_j} - \sum_i^n m_i \boldsymbol{v}_i \frac{\partial \boldsymbol{v}_i}{\partial q_j}\right\} \cdot \delta q_j, \\
&= \sum_j^N \left\{\frac{d}{dt}\frac{\mathscr{T}}{\partial \dot{q}_j} - \frac{\partial \mathscr{T}}{\partial q_j}\right\} \cdot \delta q_j. \qquad (3.55)
\end{aligned}$$

Substitute Equation (3.55) and (3.52) into Equation (3.48) , and D'Alembert equation becomes

$$\sum_j^N \left\{\frac{d}{dt}\frac{\partial \mathscr{T}}{\partial \dot{q}_j} - \frac{\partial \mathscr{T}}{\partial q_j} - Q_j\right\}\delta q_j = 0. \qquad (3.56)$$

Because the virtual displacements are independent, the intergrant in the above equation must vanish,

$$\frac{d}{dt}\frac{\partial \mathscr{T}}{\partial \dot{q}_j} - \frac{\partial \mathscr{T}}{\partial q_j} - Q_j = 0. \qquad (3.57)$$

The generalized force term Q_j in Equation (3.57) is formed by the contribution of conservative forces and nonconservative forces. The conservative forces depend only on the generalized coordinates, and can be characterized by the potential field $\mathscr{V}$. So we can write the generalized force term Q_j as the combination of conservative generalized forces Q_c and nonconservative generalized force Q_{nc} as

$$Q_j = Q_c + Q_{nc} \qquad (3.58)$$

For the conservative generalized force Q_c, the conservative force $F_i^{D_c}$ is given in terms of the potential energy as $F_i^{D_c} = -\nabla_i \mathscr{V}(\boldsymbol{r}_i, t)$. From Equation (3.52) we have

$$\begin{aligned} Q_c &= \sum_i^n F_i^{D_c} \frac{\partial \boldsymbol{r}_i}{\partial q_j}, \\ &= -\sum_i^n \frac{\partial \boldsymbol{r}_i}{\partial q_j} \nabla_i \mathscr{V}, \\ &= -\frac{\partial \mathscr{V}}{\partial q_j}. \end{aligned} \tag{3.59}$$

We so now combine Equation (3.57) , (3.58) ,and (3.59) to get

$$\frac{d}{dt}\frac{\partial \mathscr{T}}{\partial \dot{q}_j} - \frac{\partial \mathscr{T}}{\partial q_j} + \frac{\partial \mathscr{V}}{\partial q_j} - Q_{nc} = 0 \qquad (j = 1, 2, ...N)\,. \tag{3.60}$$

The Lagrange's equations for the constrained system is

$$\frac{d}{dt}\frac{\partial \mathscr{L}}{\partial \dot{q}_j} - \frac{\partial \mathscr{L}}{\partial q_j} + \frac{\partial \mathscr{V}}{\partial q_j} = Q_{nc} \qquad (j = 1, 2, ...N)\,. \tag{3.61}$$

Through out this section, the derivation of the Lagrange's equations for both the unconstrained and holonomic constrained system are carefully defined. In addition, the generalized coordinates for the Lagrange's equations are chosen to be the joint angles. In the following section, Equation (3.61) is used to model the dynamics of the robot in this thesis.

3.2.3 Robot kinetic energy in the D-H parameter convention

The robot kinetic energy is determined by the motion of each link, and $\mathscr{T}_j$ represents the kinetic energy that link j contributes. Here we only consider a robot with all revolute joints.

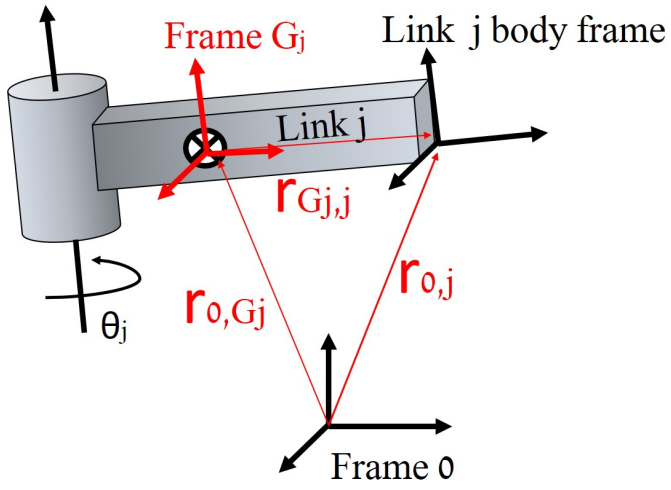


Figure 3.13: The center of the mass G_j of link j

We locate frame G_j at the mass center of link j : by definition it is parallel to body frame j. The density of m_j is given by ρ_j. Therefore, the kinetic energy $\mathscr{T}_{j-linear}$ that is associated with the translational velocity of link j becomes

$$\begin{aligned}\mathscr{T}_{j-linear} &= \frac{1}{2}\int_{Vj} \boldsymbol{v}_{0,G_j}^{0\,\intercal}\boldsymbol{v}_{0,G_j}^{0}\rho_j dV, \\ &= \frac{1}{2}m_j\boldsymbol{v}_{0,G_j}^{0\,\intercal}\boldsymbol{v}_{0,G_j}^{0}. \end{aligned} \tag{3.62}$$

To calculate the velocity of the mass center of link j, we apply the kinematic Jacobian mapping that is derived in Equation (3.18). The Jacobian matrix for the center of the mass of link j is denoted as $\boldsymbol{J}_{Vcj}$. The number of its column equals to the frame number of the link. The adventage of using the extended the Jacobian $\boldsymbol{J}_{Vcj}$ is that the matrix size of the extended form remains the same no matter how many joints are actually included in the computation. For example, for the robot with N revolute joints, if the link j center of mass is of interest, the column number 1 to j of the extended Jacobian matrix will be filled out with the original Jacobian matrix. However, column numbers $j+1$ to N are filled with zero vectors $\begin{bmatrix}0 & 0 & 0\end{bmatrix}^{\intercal}$. The form of the extended Jacobian matrix $\boldsymbol{J}_{Vcj}$ for the link j center of mass can be written as

$$
\begin{aligned}
\boldsymbol{v}^0_{0,Gj} &= \underbrace{\begin{bmatrix}\hat{\boldsymbol{z}}^0_0 \times \boldsymbol{r}^0_{0,G_1} & \hat{\boldsymbol{z}}^0_0 \times \boldsymbol{r}^0_{1,G_2} & \dots & \hat{\boldsymbol{z}}^0_{j-1} \times \boldsymbol{r}^0_{j-1,G_j}\end{bmatrix}_{3\times j}}_{original\ Jacobian\ part} \begin{bmatrix}\dot{q}_1 \\ \dot{q}_2 \\ \vdots \\ \dot{q}_j\end{bmatrix}_{j\times 1}, \\
&= \underbrace{\begin{bmatrix}\hat{\boldsymbol{z}}^0_0 \times \boldsymbol{r}^0_{0,G_1} & \hat{\boldsymbol{z}}^0_0 \times \boldsymbol{r}^0_{1,G_2} & \dots & \hat{\boldsymbol{z}}^0_{j-1} \times \boldsymbol{r}^0_{j-1,G_j} & \boldsymbol{0} & \dots & \boldsymbol{0}\end{bmatrix}_{3\times N}}_{extended\ Jacobian} \begin{bmatrix}\dot{q}_1 \\ \dot{q}_2 \\ \vdots \\ \dot{q}_j \\ \dot{q}_{j+1} \\ \vdots \\ \dot{q}_N\end{bmatrix}_{N\times 1}, \\
&= \boldsymbol{J}_{Vcj}\dot{\boldsymbol{q}}. \qquad (3.63)
\end{aligned}
$$

Note that $\boldsymbol{r}^0_{j-1,G_j}$ are the components of the vector from the origin of the body frame $j-1$ to the centre of the mass of the link j expressed in terms of frame 0. It can be recursively culculated as

$$
\begin{aligned}
\boldsymbol{r}^0_{j-1,G_j} &= \left(\boldsymbol{r}^0_{0,j} - \boldsymbol{r}^0_{0,j-1}\right) - \boldsymbol{r}^0_{j,G_j}, \\
&= \left(\boldsymbol{r}^0_{0,j} - \boldsymbol{r}^0_{0,j-1}\right) - \boldsymbol{R}^0_j \boldsymbol{r}^j_{j,G_j}. \qquad (3.64)
\end{aligned}
$$

where $\boldsymbol{r}^j_{j,G_j}$ can be easily measured from the robot `CAD` model. Now substitute Equation (3.63) into Equation (3.62). We get the kinetic energy expression for a single link j as

$$
\mathscr{T}_{j-linear} = \frac{1}{2} m_j \dot{\boldsymbol{q}}_j^\intercal \boldsymbol{J}_{Vcj}^\intercal \boldsymbol{J}_{Vcj} \dot{\boldsymbol{q}}_j. \qquad (3.65)
$$

In addition, the kinetic energy that is due to the angular velocity of the link j is expressed as

$$\begin{aligned}
\mathscr{T}_{j-angular} &= \frac{1}{2}\int_{Vj}\left(\boldsymbol{r}^0_{0,Gj}\times\boldsymbol{\omega}^0_{0,G_j}\right)^{\mathsf{T}}\cdot\left(\boldsymbol{r}^0_{0,Gj}\times\boldsymbol{\omega}^0_{0,G_j}\right)\rho dV,\\
&= \frac{1}{2}\int_{Vj}{\boldsymbol{r}^0_{0,Gj}}^{\mathsf{T}}\boldsymbol{S}^{\mathsf{T}}\left(\boldsymbol{\omega}^0_{0,Gj}\right)\boldsymbol{S}\left(\boldsymbol{\omega}^0_{0,Gj}\right)\boldsymbol{r}^0_{0,Gj}\rho dV,\\
&= \frac{1}{2}{\boldsymbol{\omega}^0_{0,Gj}}^{\mathsf{T}}\underbrace{\left\{\int_{Vj}\boldsymbol{S}^{\mathsf{T}}\left(\boldsymbol{r}^0_{0,Gj}\right)\boldsymbol{S}\left(\boldsymbol{r}^0_{0,Gj}\right)\rho dV\right\}}_{=\boldsymbol{I}^0_{mj}}\boldsymbol{\omega}^0_{0,Gj},\\
&= \frac{1}{2}{\boldsymbol{\omega}^0_{0,Gj}}^{\mathsf{T}}\boldsymbol{I}^0_{mj}\boldsymbol{\omega}^0_{0,Gj}. \qquad (3.66)
\end{aligned}$$

The second equality in Equation (3.66) holds because of the cross product property $\boldsymbol{S}\left(\boldsymbol{\omega}^0_{0,Gj}\right)\boldsymbol{r}^0_{0,Gj} = -\boldsymbol{S}\left(\boldsymbol{r}^0_{0,Gj}\right)\boldsymbol{\omega}^0_{0,Gj}$. In addition, the integral of the third equation yields the inertia tensor relative to the mass center m_j. It is expressed relative to frame 0, which makes $\boldsymbol{I}^0_{mj}$ depend on time. However, the inertia tensor is a constant matrix if it is expressed relative to the link frame. We change the coordinate expression for the inertia tensor by the transformation

$$\boldsymbol{I}^0_{mj} = \boldsymbol{R}^0_j\boldsymbol{I}^j_{mj}\boldsymbol{R}^{0\mathsf{T}}_j. \qquad (3.67)$$

Now we employ the D-H convention and the extended Jacobian matrix. The angular velocity of the center of mass of link j can be calculated based on Equation (3.18) where

$$\begin{aligned}
\boldsymbol{\omega}^0_{0,G_j} &= \underbrace{\begin{bmatrix}\hat{\boldsymbol{z}}^0_0 & \hat{\boldsymbol{z}}^0_1 & \dots & \hat{\boldsymbol{z}}^0_{j-1}\end{bmatrix}_{3\times j}}_{original\ \ Jacobian}\begin{bmatrix}\dot{q}_1\\ \dot{q}_2\\ \vdots\\ \dot{q}_j\end{bmatrix}_{j\times 1},\\
&= \underbrace{\begin{bmatrix}\hat{\boldsymbol{z}}^0_0 & \hat{\boldsymbol{z}}^0_1 & \dots & \hat{\boldsymbol{z}}^0_{j-1} & \boldsymbol{0} & \dots & \boldsymbol{0}\end{bmatrix}_{3\times N}}_{extended\ \ Jacobian}\begin{bmatrix}\dot{q}_1\\ \dot{q}_2\\ \vdots\\ \dot{q}_j\\ \dot{q}_{j+1}\\ \vdots\\ \dot{q}_N\end{bmatrix}_{N\times 1},\\
&= \boldsymbol{J}_{\omega cj}\dot{\boldsymbol{q}}. \qquad (3.68)
\end{aligned}$$

When we substitute Equation(3.67) and (3.68) into Equation (3.66), we have

$$\mathscr{T}_{j-angular} = \frac{1}{2}\dot{\boldsymbol{q}}^{\intercal}\boldsymbol{J}_{\omega cj}^{\intercal}\boldsymbol{R}_j^0\boldsymbol{I}_{mj}^j\boldsymbol{R}_j^{0\intercal}\boldsymbol{J}_{\omega cj}\dot{\boldsymbol{q}}. \tag{3.69}$$

Combine Equation (3.69) and Equation (3.65), and we can create the kinetic energy for a single link j as

$$\mathscr{T}_j = \frac{1}{2}m_j\dot{\boldsymbol{q}}_j^{\intercal}\boldsymbol{J}_{Vcj}^{\intercal}\boldsymbol{J}_{Vcj}\dot{\boldsymbol{q}}_j + \frac{1}{2}\dot{\boldsymbol{q}}^{\intercal}\boldsymbol{J}_{\omega cj}^{\intercal}\boldsymbol{R}_j^0\boldsymbol{I}_{mj}^j\boldsymbol{R}_j^{0\intercal}\boldsymbol{J}_{\omega cj}\dot{\boldsymbol{q}} \tag{3.70}$$

Within this section, the kinetic energy for a single link j has been derived. In addition, the expression for the kinetic energy employs the D-H parameter convention, which enable us to calculate systematically the value of kinetic energy at each time step.

3.2.4 Robot potential energy in the D-H parameter convention

Because the formulation of the Lagrangian contains both kinetic energy and potential energy terms, we need the expression of both before calculating the equations of motion. In the previous section, the kinetic energy expressed in terms of D-H convention has been derived. In this section we focus on the potential energy and its expression in D-H convention.

The components of the position vector of the center of the mass of link j expressed relative to frame 0 is $\boldsymbol{r}_{0,cj}^0$, and it can be calculated as

$$\begin{aligned}\boldsymbol{r}_{0,cj}^0 &= \boldsymbol{r}_{0,j}^0 - \boldsymbol{R}_j^0\boldsymbol{r}_{j,cj}^j, \\ &= \boldsymbol{H}_j^0\begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix}^{\intercal} - \boldsymbol{R}_j^0\boldsymbol{r}_{j,cj}^j.\end{aligned} \tag{3.71}$$

In this equation, $\boldsymbol{r}_{j,G_j}^j$ can be easily measured from the robot `CAD` model. Therefore, with the gravity vector expressed relative to frame 0, the potential energy of the mass center of link j is

$$\mathscr{V}_j = -m_j\boldsymbol{g}^{0\intercal}\boldsymbol{r}_{0,cj}^0, \tag{3.72}$$

where $\boldsymbol{g}^0 = \begin{bmatrix}0 & 0 & -g\end{bmatrix}^{\intercal}$. In the next section, we will begin to combine the kinetic and potential energies together in order to construct the Lagrangian for the holonomically constrained system.

3.2.5 Lagrange's equations of the robot in the D-H convention

From the previous section, we know the Lagrangian is expressed as $\mathscr{L} = \mathscr{T} - \mathscr{V}$. In addition, the total kinetic energy for the overall robot manipulator can be obtained by summing expressions such as in Equation (3.70) for the first link to the last. In this case we obtain

$$\begin{aligned}\mathscr{T}(\boldsymbol{q}, \dot{\boldsymbol{q}}) &= \sum_{j=1}^{N} \Big\{ \frac{1}{2} m_j \dot{\boldsymbol{q}}_j^{\intercal} \boldsymbol{J}_{Vcj}^{\intercal} \boldsymbol{J}_{Vcj} \dot{\boldsymbol{q}}_j + \frac{1}{2} \dot{\boldsymbol{q}}^{\intercal} \boldsymbol{J}_{\omega cj}^{\intercal} \boldsymbol{R}_j^0 \boldsymbol{I}_{mj}^j \boldsymbol{R}_j^{0\intercal} \boldsymbol{J}_{\omega cj} \dot{\boldsymbol{q}} \Big\}, \\ &= \frac{1}{2} \sum_{k=1}^{N} \sum_{j=1}^{N} m_{kj}(\boldsymbol{q}) \dot{q}_k \dot{q}_j, \\ &= \frac{1}{2} \dot{\boldsymbol{q}}^{\intercal} \boldsymbol{M}(\boldsymbol{q}) \dot{\boldsymbol{q}}. \end{aligned} \tag{3.73}$$

In these expression, $\boldsymbol{M}(\boldsymbol{q})$ is the generalized inertia matrix. Furthermore, the total potential energy of the robot manipulator is written as

$$\mathscr{V} = \sum_{j=1}^{N} -m_j \boldsymbol{g}^{0\intercal} \boldsymbol{r}_{0,cj}^0. \tag{3.74}$$

According to the Lagrange's equations (3.60) *(or* (3.61)*)* , we need to solve the following equations:

$$\begin{aligned}\frac{d}{dt}\frac{\partial \mathscr{L}}{\partial \dot{\boldsymbol{q}}} &= \frac{d}{dt}\left(\frac{\partial \mathscr{T}}{\partial \dot{q}_k} + \frac{\partial \mathscr{T}}{\partial \dot{q}_j}\right) \quad (k = 1, 2...N; j = 1, 2..., N), \\ &= 2 \cdot \frac{d}{dt}\frac{\partial \mathscr{T}}{\partial \dot{q}_k} \qquad (k = 1, 2...N), \\ &= 2 \cdot \frac{d}{dt}\Big\{ \frac{\partial}{\partial \dot{q}_k} \frac{1}{2} \sum_{k=1}^{N} \sum_{j=1}^{N} m_{kj}(\boldsymbol{q}) \dot{q}_k \dot{q}_j \Big\} \quad (k = 1, 2...N), \\ &= \frac{d}{dt}\Big\{ \sum_{j=1}^{N} m_{kj}(\boldsymbol{q}) \dot{q}_j \Big\} \qquad (k = 1, 2...N), \\ &= \sum_{j=1}^{N} m_{kj}(\boldsymbol{q}) \ddot{q}_j + \sum_{j=1}^{N} \frac{d}{dt} m_{kj}(\boldsymbol{q}) \dot{q}_j \qquad (k = 1, 2...N), \\ &= \sum_{j=1}^{N} m_{kj}(\boldsymbol{q}) \ddot{q}_j + \sum_{j=1}^{N} \sum_{w=1}^{N} \frac{\partial m_{kj}(\boldsymbol{q})}{\partial q_w} \dot{q}_w \dot{q}_j \quad (k = 1, 2...N). \end{aligned} \tag{3.75}$$

We again use Equation (3.73) to calculate the derivative of the kinetic energy with respect to joint q_k as

$$\frac{\partial \mathscr{T}}{\partial q_k} = \frac{1}{2}\sum_{k=1}^{N}\sum_{j=1}^{N}\frac{\partial m_{kj}\left(\boldsymbol{q}\right)}{\partial q_k}\dot{q}_k\dot{q}_j, \tag{3.76}$$

and the derivative of the potential energy with respect to joint angle as

$$\frac{\partial \mathscr{V}}{\partial q_k} = -\sum_{j=1}^{N} m_j \boldsymbol{g}^{\intercal}\frac{\partial \boldsymbol{r}^0_{0,cj}\left(\boldsymbol{q}\right)}{\partial q_k} \quad (k = 1, 2...N)\,. \tag{3.77}$$

Therefore, when we combine Equation (3.77) , (3.75) and (3.60) we have

$$\sum_{j=1}^{N} m_{kj}\left(\boldsymbol{q}\right)\ddot{q}_j + \sum_{j=1}^{N}\sum_{w=1}^{N}\left[\left(\frac{\partial m_{kj}}{\partial q_w} - \frac{1}{2}\frac{\partial m_{kj}}{\partial q_k}\right)\dot{q}_w\dot{q}_j\right] + \left(-\sum_{j=1}^{N} m_j \boldsymbol{g}^{\intercal}\frac{\partial \boldsymbol{r}^0_{0,cj}\left(\boldsymbol{q}\right)}{\partial q_k}\right) = Q_{nc} \quad (k = 1, 2...N)\,. \tag{3.78}$$

Equation (3.78) can be written in matrix form as

$$\boldsymbol{M}\left(\boldsymbol{q}\right)\ddot{\boldsymbol{q}} + \boldsymbol{C}\left(\boldsymbol{q}, \dot{\boldsymbol{q}}\right) + \frac{\partial \mathscr{V}}{\partial \boldsymbol{q}} = Q_{nc}. \tag{3.79}$$

It is well known that $\dot{\boldsymbol{M}} - 2\boldsymbol{C}$ is a skew symmetry matrix. During this section, the Lagrange's equations for the robot manipulator having all revolute joints is derived. It can be written in matrix form as in Equation (3.79). In addition, the Lagrange's equations of the robot follow the D-H parameter convention. This means once the D-H parameters of the robot system are defined, the corresponding Lagrange's equations can be generated systematically. Since we now know an approach to construct Lagrange's equations, the next section will focus on extracting the parameters from the `CAD` model of the robot.

3.3 Robot dynamic parameters

In our study, the ABB IRB120 robot arm has been assigned body frames on each link, and those body frame are denoted $C_1, C_2, ...C_5$. In addition, the robot end effector effectively coincides with the camera-projector unit. In addition, the camera projector unit is denoted as 6^{th} link, and the frame of the 6^{th} link is frame C. Moreover, the mass of each link from link 1 to link 5 can be obtained from the `CAD` model that is publicly available on the `ABB` website. The mass of link 6 includes the the mass of the camera C, projector P, and any

mounting fixtures. Table 3.3 summarizes the mass of each link and the location of the mass center relative to each link frame.

Table 3.3: Link mass and mass center location

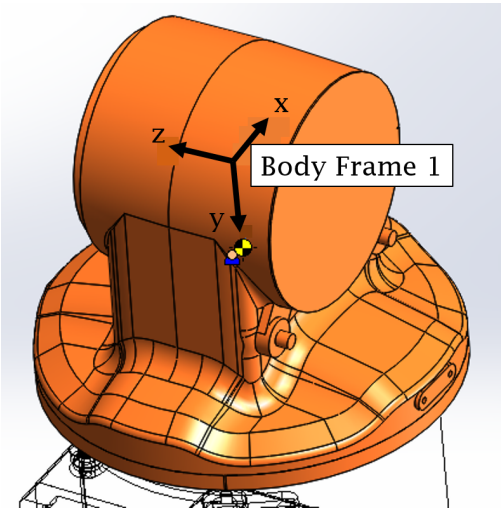
Mass Center	Mass (kg)	Center of mass location (mm)
m_{C_1}	3.07	$\boldsymbol{p}^{C_1}_{C_1,m_{C1}} = \begin{bmatrix} 0 & 51 & 0 \end{bmatrix}^T$
m_{C_2}	3.91	$\boldsymbol{p}^{C_2}_{C_2,m_{C2}} = \begin{bmatrix} -169 & -2 & -2 \end{bmatrix}^T$
m_{C_3}	2.94	$\boldsymbol{p}^{C_3}_{C_3,m_{C3}} = \begin{bmatrix} 12 & 1 & 23 \end{bmatrix}^T$
m_{C_4}	1.33	$\boldsymbol{p}^{C_4}_{C_4,m_{C4}} = \begin{bmatrix} -2 & 77 & -1 \end{bmatrix}^T$
m_{C_5}	0.55	$\boldsymbol{p}^{C_5}_{C_5,m_{C5}} = \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^T$
m_C	0.4	$\boldsymbol{p}^{C}_{C,m_C} = \begin{bmatrix} 0 & 0 & -100 \end{bmatrix}^T$

When we express the components of the inertia tensor of each link relative to their corresponding link body frame, then the inertia matrices are constant. A Solidworks® model provided by the ABB company has been used to calculate the inertia matrices of each link relative to its associated body frame. Each of the inertia matrices $I^{C_j}_{mc_j}$ below represents the inertia matrix about frame C_j with origin at the center of mass of the link j that is parallel to the frame for link j. Furthermore, the units of inertia tensor of all the entries of the matrices is (kg/m^2).

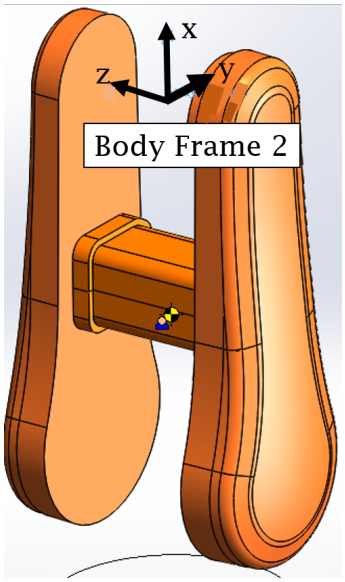
$$\boldsymbol{I}^{C_1}_{mc_1} = \begin{bmatrix} 0.02 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0.02 \end{bmatrix} \left(kg/m^2\right) \qquad \boldsymbol{I}^{C_2}_{mc_2} = \begin{bmatrix} 0.03 & 0 & 0 \\ 0 & 0.17 & 0 \\ 0 & 0 & 0.15 \end{bmatrix} \left(kg/m^2\right)$$

$$\boldsymbol{I}^{C_3}_{mc_3} = \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.02 & 0 \\ 0 & 0 & 0.01 \end{bmatrix} \left(kg/m^2\right) \qquad \boldsymbol{I}^{C_4}_{mc_4} = \begin{bmatrix} 0.07 & 0 & 0 \\ 0 & 0.07 & 0 \\ 0 & 0 & 0 \end{bmatrix} \left(kg/m^2\right)$$

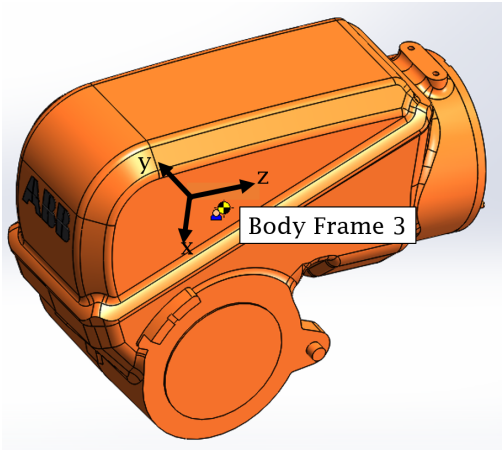
$$\boldsymbol{I}^{C_5}_{mc_5} = \begin{bmatrix} 0.05 & 0 & 0 \\ 0 & 0.05 & 0 \\ 0 & 0 & 0 \end{bmatrix} \left(kg/m^2\right) \qquad \boldsymbol{I}^{C}_{mc} = \begin{bmatrix} 0.006 & 0 & 0 \\ 0 & 0.006 & 0 \\ 0 & 0 & 0.001 \end{bmatrix} \left(kg/m^2\right)$$



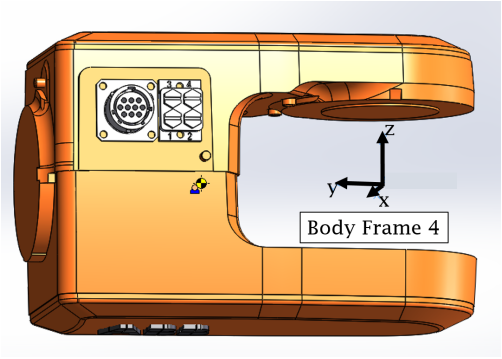
(a) Link1 Body frame and center of mass



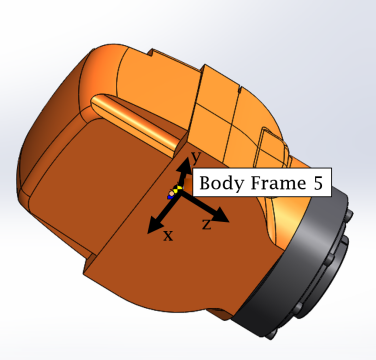
(b) Link2 Body frame and center of mass



(c) Link3 Body frame and center of mass



(d) Link4 Body frame and center of mass



(e) Link5 Body frame and center of mass

Figure 3.14: Robot linkage geometry

Chapter 4

Uncertainty models

A control goal of this thesis is to study how the uncertainties in the model will affect the system performance. In our study, the camera-projector unit on the robot end effector is mounted manually by the user. In addition, the choice of the camera on the robot can vary For example, if the projection target T is far away, the camera may not be equipped with a telephoto lens instead of using a wide angle lens. For these reasons, the projection system may have significant uncertainty in the camera mounting pose or the camera intrinsic parameter, or both.

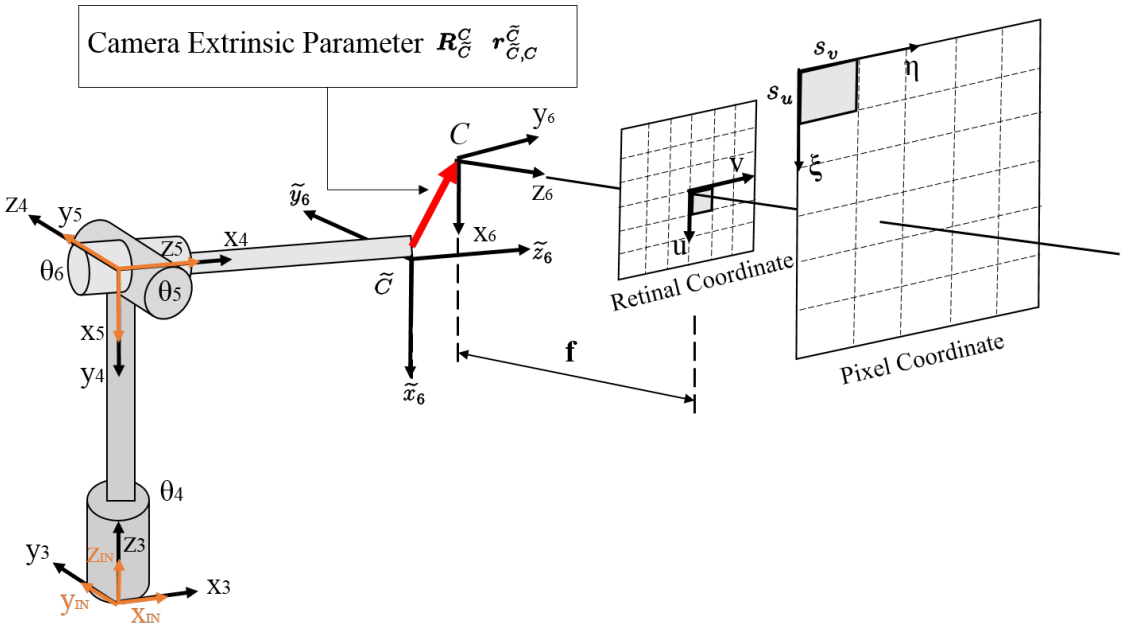


Figure 4.1: Camera intrinsic parameter s_u s_v and mounting uncertainty $\boldsymbol{R}_{\tilde{C}}^{C}$, $\boldsymbol{r}_{\tilde{C},C}^{\tilde{C}}$

4.1 Camera mounting uncertainty

Camera extrinsic parameter uncertainties arise due to mounting pose error between the true camera C and the ideal mounted camera pose $\widetilde{C}$. In general, the pose error can arise due to either displacement error, or rotation error, or both. Displacement error is measured in terms of the distance from the ideal camera position to the actual camera mounting position. That is, it is measured from the origin of frame $\widetilde{C}$ to origin of frame C. In addition, the rotational uncertainty is due to the misalignment of the orientation of $\widetilde{C}$ relative to C.

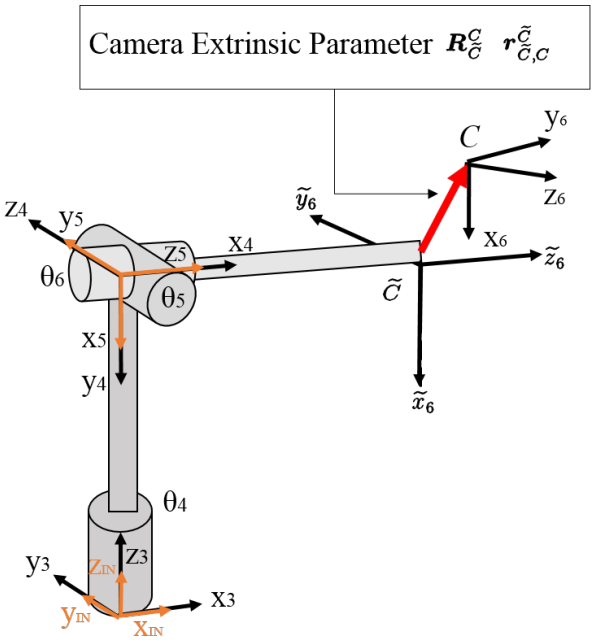


Figure 4.2: The extrinsic parameter describes the misalignment between perfect positioning camera frame C and actual mounting camera frame $\widetilde{C}$

We first neglect the position error that may arise in mounting the camera and study uncertainty only due to rotationed misalignment. Note that the Cartesian velocity of Jacobian matrix is expressed in inertial coordinates, so we first need to construct a transformation that converts $\left[\boldsymbol{v}_{I,C}^{C}\boldsymbol{\omega}_{I,C}^{C}\right]^{T}$ to $\left[\boldsymbol{v}_{I,C}^{I}\boldsymbol{\omega}_{I,C}^{I}\right]^{T}$. Because C is rigidly fixed to $\widetilde{C}$, the velocity of two points on the same rigid body must satisfy the transport equation:

$$\boldsymbol{v}_{I,C}^{I} = \boldsymbol{v}_{I,\widetilde{C}}^{I} + \boldsymbol{\omega}_{I,\widetilde{C}}^{I} \times \boldsymbol{r}_{\widetilde{C},C}^{I}. \tag{4.1}$$

To change basis,we multiply (4.1) both side with $\boldsymbol{R}_{\boldsymbol{I}}^{\boldsymbol{C}}$. We have

$$\boldsymbol{R}_{I}^{C}\boldsymbol{v}_{I,C}^{I} = \boldsymbol{R}_{I}^{C}\boldsymbol{v}_{I,\widetilde{C}}^{I} + \boldsymbol{R}_{I}^{C}\left(\boldsymbol{\omega}_{I,\widetilde{C}}^{I} \times \boldsymbol{r}_{\widetilde{C},C}^{I}\right). \tag{4.2}$$

Note that the left hand side of Equation (4.2) is $\boldsymbol{v}^C_{I,C}$. The rotation matrix on the right hand side can be decomposed as $\boldsymbol{R}^C_I = \boldsymbol{R}^C_{\widetilde{C}}\boldsymbol{R}^{\widetilde{C}}_I$, where $\boldsymbol{R}^C_{\widetilde{C}}$ is the constant orientation uncertainty of the true camera C. We rewrite (4.2) as

$$\boldsymbol{v}^C_{I,C} = \boldsymbol{R}^C_{\widetilde{C}}\boldsymbol{R}^{\widetilde{C}}_I\boldsymbol{v}^I_{I,\widetilde{C}} + \boldsymbol{R}^C_{\widetilde{C}}\boldsymbol{R}^{\widetilde{C}}_I\boldsymbol{\omega}^I_{I,\widetilde{C}} \times \boldsymbol{r}^I_{\widetilde{C},C}. \tag{4.3}$$

In this equation $\boldsymbol{r}^I_{\widetilde{C},C}$ in Equation (4.3) is a time dependent coordinate vector that describes the position error of the true camera C. To express the position uncertainty as a constant vector, we change the basis to $\widetilde{C}$ via the identity

$$\boldsymbol{r}^I_{\widetilde{C},C} = \boldsymbol{R}^I_{\widetilde{C}}\boldsymbol{r}^{\widetilde{C}}_{\widetilde{C},C}. \tag{4.4}$$

Since frame $\widetilde{C}$ does not move with respect to C, $\boldsymbol{r}^{\widetilde{C}}_{\widetilde{C},C}$ is a constant. The complete velocity transformation including uncertainty can be written as

$$\boldsymbol{v}^C_{I,C} = \boldsymbol{R}^C_{\widetilde{C}}\boldsymbol{R}^{\widetilde{C}}_I\boldsymbol{v}^I_{I,\widetilde{C}} + \boldsymbol{R}^C_{\widetilde{C}}\boldsymbol{R}^{\widetilde{C}}_I\boldsymbol{\omega}^I_{I,\widetilde{C}} \times \left(\boldsymbol{R}^I_{\widetilde{C}}\boldsymbol{r}^{\widetilde{C}}_{\widetilde{C},C}\right). \tag{4.5}$$

Since the frames C and $\widetilde{C}$ are rigidly constrained to each other, it is trivial that the origin of frame C can be seen as a fixed point on a rigid body frame $\widetilde{C}$, the angular velocity satisfies the equation:

$$\boldsymbol{\omega}^I_{I,C} = \boldsymbol{\omega}^I_{I,\widetilde{C}}. \tag{4.6}$$

If we change the basis by multiplying both side of this identity with $\boldsymbol{R}^C_I$

$$\begin{aligned}\boldsymbol{\omega}^C_{I,C} &= \boldsymbol{R}^C_I\boldsymbol{\omega}^I_{I,\widetilde{C}},\\ &= \boldsymbol{R}^C_{\widetilde{C}}\boldsymbol{R}^{\widetilde{C}}_I\boldsymbol{\omega}^I_{I,\widetilde{C}}.\end{aligned} \tag{4.7}$$

Note that in Equation (4.5) and Equation (4.7), the components of the velocity vector $\boldsymbol{v}^I_{I,\widetilde{C}}$ and angular velocity $\boldsymbol{\omega}^I_{I,\widetilde{C}}$ of an ideally mounted camera C are expressed in inertial basis. Furthermore, we can compute $\boldsymbol{v}^I_{I,\widetilde{C}}$ and $\boldsymbol{\omega}^I_{I,\widetilde{C}}$ by the Jacobian matrix $\boldsymbol{J}_q$ in Equation (3.22). Note that there is no uncertain term in the Jacobian matrix $\boldsymbol{J}_q$. We combine equations (4.5) (4.7), and factor out the rotation matrix $\boldsymbol{R}^C_{\widetilde{C}}\boldsymbol{R}^{\widetilde{C}}_{\boldsymbol{I}}$, and substitute $\boldsymbol{J}_q$ to get

$$\begin{bmatrix} \boldsymbol{v}^C_{I,C} \\ \boldsymbol{\omega}^C_{I,C} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}^C_{\widetilde{C}} \boldsymbol{R}^{\widetilde{C}}_I & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}^C_{\widetilde{C}} \boldsymbol{R}^{\widetilde{C}}_I \end{bmatrix} \begin{bmatrix} \boldsymbol{v}^I_{I,\widetilde{C}} + \boldsymbol{\omega}^I_{\widetilde{C}} \times \left(\boldsymbol{R}^I_{\widetilde{C}} \boldsymbol{r}^{\widetilde{C}}_{\widetilde{C},C} \right) \\ \boldsymbol{\omega}^I_{I,\widetilde{C}} \end{bmatrix}, \text{ or}$$

$$= \begin{bmatrix} \boldsymbol{R}^C_{\widetilde{C}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}^C_{\widetilde{C}} \end{bmatrix} \begin{bmatrix} \boldsymbol{R}^{\widetilde{C}}_I & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}^{\widetilde{C}}_I \end{bmatrix} \begin{bmatrix} \boldsymbol{v}^I_{I,\widetilde{C}} \\ \boldsymbol{\omega}^I_{I,\widetilde{C}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{R}^C_{\widetilde{C}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}^C_{\widetilde{C}} \end{bmatrix} \underbrace{\begin{bmatrix} \boldsymbol{R}^{\widetilde{C}}_I \left(\boldsymbol{\omega}^I_{I,\widetilde{C}} \times \left(\boldsymbol{R}^I_{\widetilde{C}} \boldsymbol{r}^{\widetilde{C}}_{\widetilde{C},C} \right) \right) \\ \boldsymbol{0} \end{bmatrix}}_{\mathscr{A}}. \quad (4.8)$$

In $\mathscr{A}$, since $\boldsymbol{R}^{\widetilde{C}}_I$ is a rotation matrix with $\det\left(\boldsymbol{R}^{\widetilde{C}}_I\right) = 1$, it is convenient to use the alternate expressions

$$\begin{aligned} \boldsymbol{R}^{\widetilde{C}}_I \left(\boldsymbol{\omega}^I_{I,\widetilde{C}} \times \left(\boldsymbol{R}^I_{\widetilde{C}} \boldsymbol{r}^{\widetilde{C}}_{\widetilde{C},C} \right) \right) &= \left(\boldsymbol{R}^{\widetilde{C}}_I \boldsymbol{\omega}^I_{I,\widetilde{C}} \right) \times \left(\boldsymbol{R}^{\widetilde{C}}_I \boldsymbol{R}^I_{\widetilde{C}} \boldsymbol{r}^{\widetilde{C}}_{\widetilde{C},C} \right), \\ &= -\boldsymbol{r}^{\widetilde{C}}_{\widetilde{C},C} \times \left(\boldsymbol{R}^{\widetilde{C}}_I \boldsymbol{\omega}^I_{I,\widetilde{C}} \right), or \\ &= -\boldsymbol{Skew}\left(\boldsymbol{r}^{\widetilde{C}}_{\widetilde{C},C} \right) \boldsymbol{R}^{\widetilde{C}}_I \boldsymbol{\omega}^I_{I,\widetilde{C}}. \end{aligned} \quad (4.9)$$

Note that the camera extrinsic parameters are determined by the uncertain rotation matrix $\boldsymbol{R}^C_{\widetilde{C}}$ and the uncertain displacement vector $\boldsymbol{r}^{\widetilde{C}}_{\widetilde{C},C}$. Using equations (3.22), (3.35), (4.8) and (4.9), the mapping from the derivatives of the generalized coordinates to the pixel velocity is now written in the form

$$\begin{bmatrix} \dot{\zeta}_1 \\ \dot{\eta}_1 \\ \dot{\zeta}_2 \\ \dot{\eta}_2 \end{bmatrix} = \hat{\boldsymbol{J}}_1 \dot{\boldsymbol{q}}_{\widetilde{C}} + \hat{\boldsymbol{J}}_2 \dot{\boldsymbol{q}}_{\widetilde{C}}. \quad (4.10)$$

In this equation $\hat{\boldsymbol{J}}_1$ and $\hat{\boldsymbol{J}}_2$ are given by the expressions

$$\begin{aligned} \hat{\boldsymbol{J}}_1 &= \hat{\boldsymbol{J}}_s \begin{bmatrix} \boldsymbol{R}^C_{\widetilde{C}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}^C_{\widetilde{C}} \end{bmatrix} \begin{bmatrix} \boldsymbol{R}^{\widetilde{C}}_I & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}^{\widetilde{C}}_I \end{bmatrix} \boldsymbol{J}_q, \text{ and} \\ \hat{\boldsymbol{J}}_2 &= \hat{\boldsymbol{J}}_s \begin{bmatrix} \boldsymbol{R}^C_{\widetilde{C}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}^C_{\widetilde{C}} \end{bmatrix} \begin{bmatrix} \boldsymbol{diag}\,(1,1,1) \\ \boldsymbol{0}_{3\times 3} \end{bmatrix}_{6\times 3} \left\{ -\boldsymbol{Skew}\left(\boldsymbol{r}^{\widetilde{C}}_{\widetilde{C},C} \right) \boldsymbol{R}^{\widetilde{C}}_I \begin{bmatrix} \hat{\boldsymbol{z}}^I_{C3} & \hat{\boldsymbol{z}}^I_{C4} & \hat{\boldsymbol{z}}^I_{C5} \end{bmatrix}_{3\times 3} \right\}. \end{aligned} \quad (4.11)$$

Where $\hat{\boldsymbol{J}}_s$ is $\boldsymbol{J}_s$, but with terms s_u and s_v to be uncertain. It is clear that both the uncertain rotation matrix $\boldsymbol{R}^C_{\widetilde{C}}$ and the uncertain displacement vector $\boldsymbol{r}^C_{\widetilde{C},C}$ contribute to $\hat{\boldsymbol{J}}_2$, but only $\boldsymbol{R}^C_{\widetilde{C}}$ influences $\hat{\boldsymbol{J}}_1$.

4.2 Camera intrinsic uncertainty

As we mentioned in the previous chapter, we employ the pinhole camera model in this thesis. In addition, in our study we only consider the pixel scaling factors s_u and s_v. Also note that our pixel plane model does not include pixel shearing distortion.

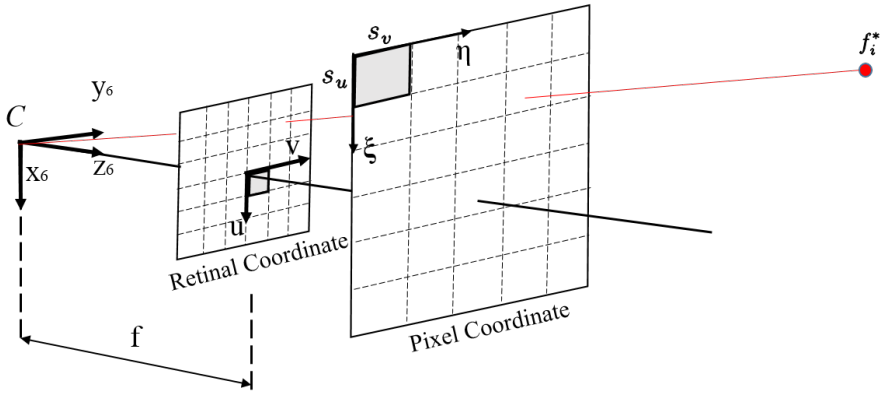


Figure 4.3: Camera focal length and pixel coordinate scaling factor are the uncertainties in the camera perspective projection model

Through this chapter, the uncertainties in our system are clearly defined. The uncertainties include the orientation of the camera $\boldsymbol{R}_{\widetilde{C}}^{C}$ and the pixel space scaling factor s_u and s_v. In the next chapter, we will illustrate how to build an adaptive controller that can compensate for these uncertainties.

Chapter 5

Controller design

In this thesis, the controller that we derive must be able to accommodate the projector and camera mounting error and the uncertain camera intrinsic parameters. In fact, since the projector-camera unit should be first mounted as accurately as possible by the user, the dynamic uncertainty between the ideal end effector pose and actual pose is neglected. However, the end effector kinematic uncertainty still plays an important role since misalignment induces errors that are more pronounced as we increase the distance between target and the projector. For instance, the image projection error on a target three feet distant is less than the projection error on the target that is 100 feet distant.

5.1 Adaptive controller

The purpose of the adaptive controller in this thesis is to compensate for the kinematic uncertainty , that is, the uncertainties occur in the Jacobian matrix $\boldsymbol{J}$. Since those uncertain parameters are constant and can be factored from the Jacobain matrix, we say the kinematic Jacobian $\boldsymbol{J}$ is linear in parameters. In this case we are able to write the Jacobian in regressor form. In addition, the learning law in the controller will update these uncertain parameters on time based on the previous state. The updated parameters are not guaranteed to match the unknown actual parameters, unless the system is persistantly excited. A system is persistently excited when the input signal is rich enough and contains many frequencies. However, a controller should guarantee the system output tracking error converges to zero, even though the estimated parameters may not converge to the true ones.

Within the framework of image based servo control, the output error $\boldsymbol{e}_s$ of the system is the image space error between the desired feature $\boldsymbol{s}^*$ and the current feature $\boldsymbol{s}$. The control input to the system is the torque that is applied to each robot link. The applied torque is calculated based on the estimation of the kinematic Jacobian matrix. In the next section, we will illustrate how the kinematic Jacobian can be cast in the controller regressor form.

5.1.1 System regression

Base on Equation (4.10) and Equation (4.11), the complete system kinematic Jacobian matrix with uncertainties is

$$\begin{bmatrix} \dot{\zeta}_1 \\ \dot{\eta}_1 \\ \dot{\zeta}_2 \\ \dot{\eta}_2 \end{bmatrix} = \hat{\boldsymbol{J}}_1 \dot{\boldsymbol{q}}_{\widetilde{C}} + \hat{\boldsymbol{J}}_2 \dot{\boldsymbol{q}}_{\widetilde{C}}.$$

For the case that when we neglect the displacement uncertainty, but only consider the uncertain orientation of the projector-camera unit, the term $\begin{bmatrix} \hat{\boldsymbol{z}}^I_{C3} & \hat{\boldsymbol{z}}^I_{C4} & \hat{\boldsymbol{z}}^I_{C5} \end{bmatrix}_{3x3}$ becomes zero, so $\hat{\boldsymbol{J}}_2$ vanishes. In order to make the regression expression more compact, we rearrange the order of the pixel error and write

$$\dot{\hat{\boldsymbol{s}}} = \begin{bmatrix} \dot{\zeta}_1 \\ \dot{\zeta}_2 \\ \dot{\eta}_1 \\ \dot{\eta}_2 \end{bmatrix} = \hat{\mathbb{J}} \dot{\boldsymbol{q}}_{\widetilde{C}} \tag{5.1}$$

Furthermore, in order to write Equation (5.1) in regressor form, we need the constant uncertain parameters to form a vector. Note that here we set the focal length to be nominal value $f = 1$, and let the camera pixel space scaling factors s_v, s_u to represent all of the scaling effect. By inspecting the structure of the kinematic Jacobian, we find that

$$\dot{\boldsymbol{s}} = \hat{\mathbb{J}} \dot{\boldsymbol{q}}_{\widetilde{C}} = \boldsymbol{Y}_k \left(\boldsymbol{q}_{\widetilde{C}}, \dot{\boldsymbol{q}}_{\widetilde{C}} \right) \hat{\boldsymbol{\theta}}_k \ , \tag{5.2}$$

where $\boldsymbol{Y}_k \left(\boldsymbol{q}_{\widetilde{C}}, \dot{\boldsymbol{q}}_{\widetilde{C}} \right)$ is a matrix with all known terms, and $\hat{\boldsymbol{\theta}}_k$ is the vector that contains all of the constant uncertain parameters. When we write $\hat{\boldsymbol{\theta}}_k$ out explicitly we have

$$\hat{\boldsymbol{\theta}}_k = \begin{bmatrix} s_u r_{11} & s_u r_{12} & s_u r_{13} & \dots & s_u r_{ij} & s_v r_{11} & s_v r_{12} & s_v r_{13} & \dots & s_v r_{ij} \end{bmatrix}^\intercal \ . \tag{5.3}$$

In Equation (5.3) s_u and s_v are the pixel scaling factors, and r_{ij} are the elements of the uncertain orientation matrix $\boldsymbol{R}^C_{\widetilde{C}}$. We write $\boldsymbol{R}^C_{\widetilde{C}}$ as

$$\boldsymbol{R}^C_{\widetilde{C}} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \ . \tag{5.4}$$

5.1.2 The stability proof for the calibration controller

The equations of motion of the robotic manipulator that includes the camera and projector can be written in terms of an 3×3 symmetric positive definite generalized mass matrix $\boldsymbol{M}_{\widetilde{C}}\left(\boldsymbol{q}_{\widetilde{C}}\right)$, an 3×3 nonlinear matrix of centripetal and Coriolis matrix $\boldsymbol{C}_{\widetilde{C}}\left(\boldsymbol{q}_{\widetilde{C}}, \dot{\boldsymbol{q}}_{\widetilde{C}}\right)$, and potential energy $\mathscr{V}$,

$$\boldsymbol{M}_{\widetilde{C}}\left(\boldsymbol{q}_{\widetilde{C}}\right) \ddot{\boldsymbol{q}}_{\widetilde{C}} + \boldsymbol{C}_{\widetilde{C}}\left(\boldsymbol{q}_{\widetilde{C}}, \dot{\boldsymbol{q}}_{\widetilde{C}}\right) \dot{\boldsymbol{q}}_{\widetilde{C}} + \frac{\partial \mathscr{V}}{\partial \boldsymbol{q}_{\widetilde{C}}} = \boldsymbol{\tau}_{\widetilde{\boldsymbol{C}}} \ . \tag{5.5}$$

In this paper we choose the nonlinear feedback control law to be given by $\boldsymbol{\tau}_{\widetilde{C}}$ where

$$\boldsymbol{\tau}_{\widetilde{C}} = -\hat{\mathbb{J}}^{\intercal} \boldsymbol{K}_p \boldsymbol{e}_s - \boldsymbol{K}_v \dot{\boldsymbol{q}}_{\widetilde{C}} + \frac{\partial \mathscr{V}}{\partial \boldsymbol{q}_{\widetilde{C}}} \tag{5.6}$$

In this equation $\mathbb{J}$ is the re-ordered system kinematic Jacobian matrix, $\boldsymbol{e}_s$ is the 4×1 tracking error, $\boldsymbol{K}_p$ is a 4×4 gain matrix, and $\boldsymbol{K}_v$ is a 3×3 gain matrix.We choose the parameter update law $\dot{\hat{\boldsymbol{\theta}}}_k$ that is based on the classical projection operator

$$\operatorname{Proj}\left(\boldsymbol{\theta}_k, \dot{\hat{\boldsymbol{\theta}}}_k\right) = \begin{cases} \boldsymbol{L}_k \boldsymbol{Y}_k^{\intercal} \boldsymbol{K}_p \boldsymbol{e}_s & \text{if } g\left(\boldsymbol{\theta}_k\right) \leq 0 \\ \boldsymbol{L}_k \boldsymbol{Y}_k^{\intercal} \boldsymbol{K}_p \boldsymbol{e}_s & \text{if } g\left(\boldsymbol{\theta}_k\right) > 0 \text{ and } \nabla g^{\intercal} \dot{\hat{\boldsymbol{\theta}}}_k \leq 0 \\ \dot{\hat{\boldsymbol{\theta}}}_k - \frac{\nabla g}{\|\nabla g\|} \langle \frac{\nabla g}{\|\nabla g\|}, \dot{\hat{\theta}}_k \rangle g\left(\boldsymbol{\theta}_k\right) & \text{if } g\left(\boldsymbol{\theta}_k\right) > 0 \text{ and } \nabla g^{\intercal} \dot{\hat{\boldsymbol{\theta}}}_k > 0 \end{cases} \tag{5.7}$$

We will show that with the definition of the task space variable as

$$\boldsymbol{s}(t) = \begin{bmatrix} h_1(\Phi) \\ h_1(\Phi) \\ \vdots \\ h_n(\Phi) \end{bmatrix} = \begin{bmatrix} \zeta_1 \\ \zeta_2 \\ \eta_1 \\ \eta_2 \end{bmatrix} \ , \tag{5.8}$$

we have the convergence property

$$\lim_{t \to \infty} \boldsymbol{s}(t) \longrightarrow \boldsymbol{s}^* . \tag{5.9}$$

Proof:

From Equations (4.10), (4.11) and the assumptions above, the estimated kinematic Jacobian is given by

$$\begin{bmatrix} \dot{\zeta}_1 \\ \dot{\eta}_1 \\ \dot{\zeta}_2 \\ \dot{\eta}_2 \end{bmatrix} = \hat{\boldsymbol{J}}_1 \dot{\boldsymbol{q}}_{\widetilde{C}} \ . \tag{5.10}$$

We rearrange the pixel coordinate velocity and write the estimated Jacobian in regressor form as

$$\dot{\hat{\boldsymbol{s}}} = \begin{bmatrix} \dot{\zeta}_1 \\ \dot{\zeta}_2 \\ \dot{\eta}_1 \\ \dot{\eta}_2 \end{bmatrix} = \hat{\mathbb{J}} \dot{\boldsymbol{q}}_{\widetilde{C}} = \boldsymbol{Y}_k \hat{\boldsymbol{\theta}}_k \ . \tag{5.11}$$

where $\hat{\boldsymbol{\theta}}_k$ contains the estimated parameters. We next define the pixel error as $\boldsymbol{e}_s = \boldsymbol{s} - \boldsymbol{s}^*$ and parameter error as $\widetilde{\boldsymbol{\theta}}_k = \boldsymbol{\theta_k} - \hat{\boldsymbol{\theta}}_{\boldsymbol{k}}$. Finally we define the candidate Lyapnov function as

$$V = \frac{1}{2} \dot{\boldsymbol{q}}_{\widetilde{C}}^{\intercal} \boldsymbol{M}_{\widetilde{C}} \left(\boldsymbol{q}_{\widetilde{C}} \right) \dot{\boldsymbol{q}}_{\widetilde{C}} + \frac{1}{2} \widetilde{\boldsymbol{\theta}}_k^{\intercal} \boldsymbol{L}_k^{\dashv} \widetilde{\boldsymbol{\theta}}_k + \frac{1}{2} \boldsymbol{e}_s^{\intercal} \boldsymbol{K}_p \boldsymbol{e}_s. \tag{5.12}$$

This function is smooth, $V(\boldsymbol{0}) = 0$, and it is positive definite. Now when we take time derivative of V along the trajectory of the system, we find that

$$\begin{aligned} \frac{\partial V}{\partial t} &= \dot{\boldsymbol{q}}_{\widetilde{C}}^{\intercal} \boldsymbol{M}_{\widetilde{C}} \left(\boldsymbol{q}_{\widetilde{C}} \right) \ddot{\boldsymbol{q}}_{\widetilde{C}}^{\intercal} + \frac{1}{2} \dot{\boldsymbol{q}}_{\widetilde{C}}^{\intercal} \dot{\boldsymbol{M}}_{\widetilde{C}} \left(\boldsymbol{q}_{\widetilde{C}} \right) \dot{\boldsymbol{q}}_{\widetilde{C}} + \widetilde{\boldsymbol{\theta}}_k^{\intercal} \boldsymbol{L}_k^{\dashv} \dot{\widetilde{\boldsymbol{\theta}}}_k + \boldsymbol{e}_s^{\intercal} \boldsymbol{K}_p \dot{\boldsymbol{e}}_s \ , \text{ or} \\ &= \dot{\boldsymbol{q}}_{\widetilde{C}}^{\intercal} \left(\boldsymbol{\tau}_{\widetilde{C}} - \frac{\partial \mathscr{V}}{\partial \boldsymbol{q}_{\widetilde{C}}} \right) + \widetilde{\boldsymbol{\theta}}_k^{\intercal} \boldsymbol{L}_k^{\dashv} \dot{\widetilde{\boldsymbol{\theta}}}_k + \boldsymbol{e}_s^{\intercal} \boldsymbol{K}_p \dot{\boldsymbol{e}}_s + \frac{1}{2} \dot{\boldsymbol{q}}_{\widetilde{C}}^{\intercal} \underbrace{\left(\dot{\boldsymbol{M}}_{\widetilde{C}} - 2 \boldsymbol{C}_{\widetilde{C}} \right)}_{skew \ \ symmetric} \dot{\boldsymbol{q}}_{\widetilde{C}} \ . \end{aligned} \tag{5.13}$$

The last term of Equation (5.13) equals zero because of the skew symmetric matrix property. The introduction of the feedback control law $\boldsymbol{\tau}_{\widetilde{C}}$ yields the result

$$\begin{aligned} \frac{\partial V}{\partial t} &= \left(-\dot{\boldsymbol{q}}_{\widetilde{C}}^{\intercal} \hat{\mathbb{J}}^{\intercal} \boldsymbol{K}_p \boldsymbol{e}_s - \dot{\boldsymbol{q}}_{\widetilde{C}}^{\intercal} \boldsymbol{K}_v \dot{\boldsymbol{q}}_{\widetilde{C}} \right) + \widetilde{\boldsymbol{\theta}}_k^{\intercal} \boldsymbol{L}_k^{\dashv} \dot{\widetilde{\boldsymbol{\theta}}}_k + \boldsymbol{e}_s^{\intercal} \boldsymbol{K}_p \dot{\boldsymbol{e}}_s \ , \text{ and} \\ &= -\dot{\hat{\mathbb{S}}}^{\intercal} \boldsymbol{K}_p \boldsymbol{e}_s - \dot{\boldsymbol{q}}_{\widetilde{C}}^{\intercal} \boldsymbol{K}_v \dot{\boldsymbol{q}}_{\widetilde{C}} + \widetilde{\boldsymbol{\theta}}_k^{\intercal} \boldsymbol{L}_k^{\dashv} \dot{\widetilde{\boldsymbol{\theta}}}_k + \boldsymbol{e}_s^{\intercal} \boldsymbol{K}_p \dot{\boldsymbol{e}}_s. \end{aligned} \tag{5.14}$$

Note that since the target $\boldsymbol{s}^*$ is a constant vector, we have $\dot{\boldsymbol{e}}_s = \dot{\boldsymbol{s}} - \dot{\boldsymbol{s}^*} = \dot{\boldsymbol{s}}$. The negative semidefiniteness of the Lyapnov function then follows when we substitute the parameter update law $\dot{\hat{\boldsymbol{\theta}}}_k$ into the equation we then see that

$$\begin{aligned}
\frac{\partial V}{\partial t} &= -\dot{\hat{\boldsymbol{s}}}^\intercal \boldsymbol{K}_p \boldsymbol{e}_s - \dot{\boldsymbol{q}}_{\widetilde{C}}^\intercal \boldsymbol{K}_v \dot{\boldsymbol{q}}_{\widetilde{C}} + \boldsymbol{e}_s^\intercal \boldsymbol{K}_p \dot{\boldsymbol{s}} + \boldsymbol{e}_s^\intercal \boldsymbol{K}_p \dot{\boldsymbol{e}}_s + \widetilde{\boldsymbol{\theta}}_k^\intercal \boldsymbol{Y}_k^\intercal \boldsymbol{K}_p \boldsymbol{e}_s, \\
&= -\dot{\hat{\boldsymbol{s}}}^\intercal \boldsymbol{K}_p \boldsymbol{e}_s - \dot{\boldsymbol{q}}_{\widetilde{C}}^\intercal \boldsymbol{K}_v \dot{\boldsymbol{q}}_{\widetilde{C}} + \boldsymbol{e}_s^\intercal \boldsymbol{K}_p \dot{\boldsymbol{s}} + \boldsymbol{e}_s^\intercal \boldsymbol{K}_p \dot{\boldsymbol{e}}_s - \left(\boldsymbol{\theta}_k - \hat{\boldsymbol{\theta}}_k\right)^\intercal \boldsymbol{Y}_k^\intercal \boldsymbol{K}_p \boldsymbol{e}_s, \\
&= -\dot{\hat{\boldsymbol{s}}}^\intercal \boldsymbol{K}_p \boldsymbol{e}_s - \dot{\boldsymbol{q}}_{\widetilde{C}}^\intercal \boldsymbol{K}_v \dot{\boldsymbol{q}}_{\widetilde{C}} + \boldsymbol{e}_s^\intercal \boldsymbol{K}_p \dot{\boldsymbol{s}} + \boldsymbol{e}_s^\intercal \boldsymbol{K}_p \dot{\boldsymbol{e}}_s - \left(\boldsymbol{Y}_k \boldsymbol{\theta}_k\right)^\intercal \boldsymbol{K}_p \boldsymbol{e}_s + \left(\boldsymbol{Y}_k \hat{\boldsymbol{\theta}}_k\right)^\intercal \boldsymbol{K}_p \boldsymbol{e}_s, \\
&= -\dot{\hat{\boldsymbol{s}}}^\intercal \boldsymbol{K}_p \boldsymbol{e}_s - \dot{\boldsymbol{q}}_{\widetilde{C}}^\intercal \boldsymbol{K}_v \dot{\boldsymbol{q}}_{\widetilde{C}} + \boldsymbol{e}_s^\intercal \boldsymbol{K}_p \dot{\boldsymbol{s}} + \boldsymbol{e}_s^\intercal \boldsymbol{K}_p \dot{\boldsymbol{e}}_s - \dot{\boldsymbol{s}}^\intercal \boldsymbol{K}_p \boldsymbol{e}_s + \dot{\hat{\boldsymbol{s}}}^\intercal \boldsymbol{K}_p \boldsymbol{e}_s \boldsymbol{K}_p \boldsymbol{e}_s \text{ , and} \\
&= -\dot{\boldsymbol{q}}_{\widetilde{C}}^\intercal \boldsymbol{K}_v \dot{\boldsymbol{q}}_{\widetilde{C}} \text{ .}
\end{aligned} \tag{5.15}$$

The Lyapnov function is positive definite and is noincreasing and the Lyapnov function has a positive invariant set, where $\dot{V} = 0$. Beacuse the time derivative of the Lyapnov function along the system trajectory is zero, we have $\|\dot{\boldsymbol{q}}_{\widetilde{C}}\|^2 = 0$, which indicates that $\dot{\boldsymbol{q}}_{\widetilde{C}} = \boldsymbol{0}$, and further indicates that $\ddot{\boldsymbol{q}}_{\widetilde{C}} = \boldsymbol{0}$. The closed loop robot equations of motion can be obtained by combining Equation (5.5) and Equation (5.6), we have

$$\boldsymbol{M}_{\widetilde{C}}\left(\boldsymbol{q}_{\widetilde{C}}\right) \cancelto{0}{\ddot{\boldsymbol{q}}_{\widetilde{C}}} + \boldsymbol{C}_{\widetilde{C}}\left(\boldsymbol{q}_{\widetilde{C}}, \dot{\boldsymbol{q}}_{\widetilde{C}}\right) \cancelto{0}{\dot{\boldsymbol{q}}_{\widetilde{C}}} + \hat{\mathbb{J}}^\intercal \boldsymbol{K}_p \boldsymbol{e}_s + \boldsymbol{K}_v \cancelto{0}{\dot{\boldsymbol{q}}_{\widetilde{C}}} = 0 \text{ .} \tag{5.16}$$

For the case where the gain matrix $\boldsymbol{K}_p = \beta \boldsymbol{I}_{4\times 4}$ *(β is a positive scalar, $\boldsymbol{I}$ is a identity matrix)*, and $\hat{\mathbb{J}}^\intercal$ is full column rank, we will have $\boldsymbol{e}_s = \boldsymbol{0}$ in this positive invariant set. In sum, the Lyapnov equation in the proof is noincreasing, so the system trajectories will eventually enter this positive invariant set. Once the trajectories enter this set, the pixel error $\boldsymbol{e}_s$ of the system becomes zero.

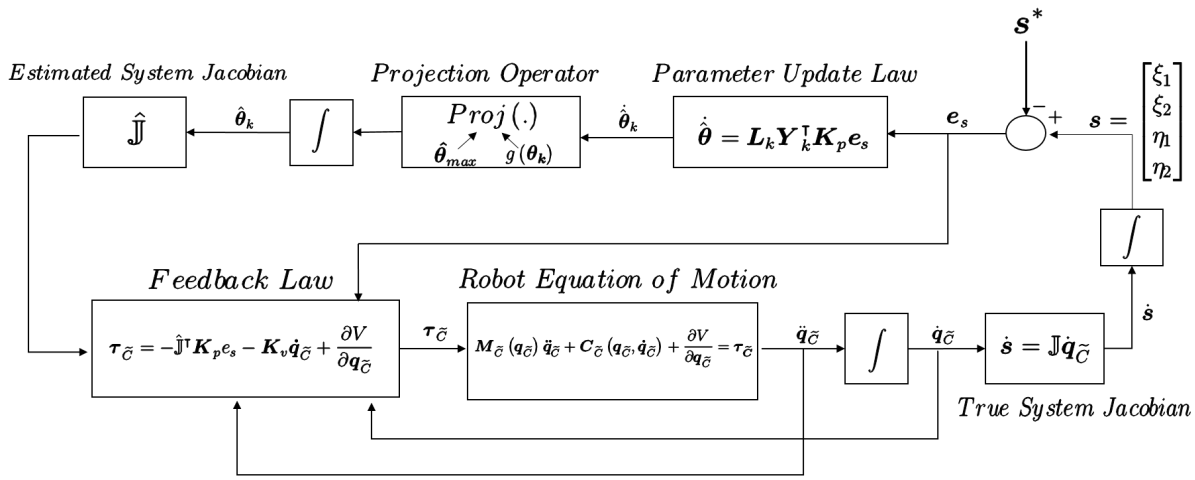


Figure 5.1: The block diagram for implementing the calibration controller

Figure 5.1 depicts a block diagram that summarizes the closed loop system. The actual robot dynamic and kinematic model are contained in blocks *Robot Equation of Motion* and *True System Jacobian*. In addition, we measure the image feature for each time step within the camera pixel space, and it is denoted as $\boldsymbol{s}$. The measured current image feature $\boldsymbol{s}$ is subtracted with the ideal image feature $\boldsymbol{s}^*$ which gives the pixel error $\boldsymbol{e}_s$. The pixel error $\boldsymbol{e}_s$ is input into the controller which contains blocks *Parameter Update Law*, *Projection Operator*, *Estimated System Jacobian*, and *Feedback Law*. In the *Parameter Update Law* block, gain matrix $\boldsymbol{L}_k$, $\boldsymbol{K}_p$, and the known part of the kinematic regression $\boldsymbol{Y}_k\left(\boldsymbol{q}_{\widetilde{C}}, \dot{\boldsymbol{q}}_{\widetilde{C}}\right)$ are needed. Also, the *Projection Operator* block is employed so that the parameter estimation remains bounded. The *Estimated System Jacobian* block is constructed in real time based on the estimated parameter $\dot{\hat{\boldsymbol{\theta}}}_k$ at each time step. The output from the controller is the computed torque $\boldsymbol{\tau}_{\widetilde{C}}$. The computed torque $\boldsymbol{\tau}_{\widetilde{C}}$ is applied to the robot link, thereby closing the control loop.

5.1.3 The stability proof for the tracking controller

Refer to the same system equation of motion in Equation (5.5), and apply the nonlinear feedback control law $\boldsymbol{\tau}_{\widetilde{C}}$ given as

$$\boldsymbol{\tau}_{\widetilde{C}} = \left(\boldsymbol{M}\dot{\hat{\mathbb{J}}}^\dagger + \boldsymbol{C}\hat{\mathbb{J}}^\dagger\right)\dot{\boldsymbol{s}}_r + \boldsymbol{M}\hat{\mathbb{J}}^\dagger\ddot{\boldsymbol{s}}_r + \frac{\partial \mathscr{V}}{\partial \boldsymbol{q}_{\widetilde{C}}} - \hat{\mathbb{J}}^\intercal \boldsymbol{K}_p \boldsymbol{e}_s \ . \tag{5.17}$$

By using the same parameter update law (5.7), we have the convergence property

$$\lim_{t\to\infty} \boldsymbol{s}(t) \longrightarrow \boldsymbol{s}^*(t) \, . \tag{5.18}$$

Proof:

Similar to the calibration controller, the task space variable is denoted as $\boldsymbol{s}$. However, the desired task space variable $\boldsymbol{s}^*$ is now a time dependent trajectory. In addition, define the tracking error as $\boldsymbol{e}_s = \boldsymbol{s} - \boldsymbol{s}^*$. The task space tracking error is denoted as $\hat{\boldsymbol{e}}_s = \hat{\boldsymbol{s}} - \hat{\boldsymbol{s}^*}$, where $\dot{\hat{\boldsymbol{s}}} = \hat{\mathbb{J}}\dot{\boldsymbol{q}}_{\widetilde{C}}$. We next combine two kinds of error: $\boldsymbol{e}_s$ and $\hat{\boldsymbol{e}}_s$. We define a vector $\boldsymbol{v}_s$ in the form

$$\boldsymbol{v}_s = \dot{\hat{\boldsymbol{e}}}_s + \alpha \boldsymbol{e}_s = \dot{\hat{\boldsymbol{s}}} - \left\{ \dot{\boldsymbol{s}}^* - \alpha \left(\boldsymbol{s} - \boldsymbol{s}^* \right) \right\} . \tag{5.19}$$

Note that α is a positive scalar. In this equation, we define a reference signal as

$$\dot{\boldsymbol{s}}_r = \dot{\boldsymbol{s}}^* - \alpha \left(\boldsymbol{s} - \boldsymbol{s}^* \right) \tag{5.20}$$

where $\dot{\boldsymbol{s}}^*$ $\boldsymbol{s}^*$ α are known, and $\boldsymbol{s}$ is measurable via the camera. Now substitude $\dot{\boldsymbol{s}}_r$ back into Equation (5.19). We have

$$\boldsymbol{v}_s = \dot{\hat{\boldsymbol{s}}} - \dot{\boldsymbol{s}}_r \ . \tag{5.21}$$

For the reference signal $\boldsymbol{s}_r$, we can calculate its corresponding joint variable derivatives $\dot{\boldsymbol{q}}_r$ based on the estimated kinematic Jacobian $\hat{\mathbb{J}}$ as

$$\dot{\boldsymbol{s}}_r = \hat{\mathbb{J}}\dot{\boldsymbol{q}}_r \tag{5.22}$$

After defining the reference derivatives $\dot{\boldsymbol{q}}_r$, we combine it with the measured robot joint velocity $\dot{\boldsymbol{q}}_{\widetilde{C}}$ to construct the sliding vector $\boldsymbol{v}_q$ in the expression

$$\boldsymbol{v}_q = \dot{\boldsymbol{q}}_{\widetilde{C}} - \dot{\boldsymbol{q}}_r. \tag{5.23}$$

According to the identities $\dot{\boldsymbol{q}}_{\widetilde{C}} = \hat{\mathbb{J}}^{\dagger}\dot{\hat{\boldsymbol{s}}}$ and $\dot{\boldsymbol{q}}_r = \hat{\mathbb{J}}^{\dagger}\dot{\boldsymbol{s}}_r$, vectors $\boldsymbol{v}_s$ and $\boldsymbol{v}_q$ in Equation (5.23) and Equation (5.21) satisfy the relationship

$$\boldsymbol{v}_q = \hat{\mathbb{J}}^{\dagger}\boldsymbol{v}_s \tag{5.24}$$

We next substitude the sliding vector in Equation (5.23) into the robot equations of motion (5.5). We subsquently have

$$\boldsymbol{M}\dot{\boldsymbol{v}}_q + \boldsymbol{C}\boldsymbol{v}_q + \boldsymbol{M}\ddot{\boldsymbol{q}}_r + \boldsymbol{C}\dot{\boldsymbol{q}}_r + \frac{\partial \mathscr{V}}{\partial \boldsymbol{q}_{\widetilde{C}}} = \boldsymbol{\tau}_{\widetilde{\boldsymbol{C}}}. \tag{5.25}$$

To make the image space properties evident in equations of motion, we substitude $\dot{\boldsymbol{q}}_r = \hat{\mathbb{J}}^{\dagger}\dot{\boldsymbol{s}}_r$ into Equation (5.25). We finally obtain

$$\boldsymbol{M}\dot{\boldsymbol{v}}_q + \boldsymbol{C}\boldsymbol{v}_q + \left(\boldsymbol{M}\dot{\hat{\mathbb{J}}}^{\dagger} + \boldsymbol{C}\hat{\mathbb{J}}^{\dagger}\right)\dot{\boldsymbol{s}}_r + \boldsymbol{M}\hat{\mathbb{J}}^{\dagger}\ddot{\boldsymbol{s}}_r + \frac{\partial \mathscr{V}}{\partial \boldsymbol{q}_{\widetilde{C}}} = \boldsymbol{\tau}_{\widetilde{\boldsymbol{C}}}\ . \tag{5.26}$$

Now we propose a Lyapnov function candidate in the form

$$V = \frac{1}{2}\boldsymbol{v}_q^{\intercal}\boldsymbol{M}\boldsymbol{v}_q + \frac{1}{2}\widetilde{\boldsymbol{\theta}}_k^{\intercal}\boldsymbol{L}^{\dashv}\widetilde{\boldsymbol{\theta}} + \frac{1}{2}\boldsymbol{e}_s^{\intercal}\boldsymbol{K}_p\boldsymbol{e}_s. \tag{5.27}$$

When we take the time derivative of the Lyapnov function along the system trajectories, we get

$$\frac{\partial V}{\partial t} = \boldsymbol{v}_q^{\intercal}\boldsymbol{M}\dot{\boldsymbol{v}}_q + \frac{1}{2}\boldsymbol{v}_q^{\intercal}\dot{\boldsymbol{M}}\boldsymbol{v}_q + \widetilde{\boldsymbol{\theta}}_k^{\intercal}\boldsymbol{L}_k^{\dashv}\dot{\boldsymbol{\theta}}_k + \boldsymbol{e}_s^{\intercal}\boldsymbol{K}_p\dot{\boldsymbol{e}}_s. \tag{5.28}$$

Replace $\boldsymbol{M}\dot{\boldsymbol{v}}_q$ by Equation (5.26) and since $\dot{\boldsymbol{M}} - 2\boldsymbol{C}$ can be proven to be skew symmetric, Equation (5.28) now becomes

$$\frac{\partial V}{\partial t} = \boldsymbol{v}_q^{\intercal}\left\{\boldsymbol{\tau}_{\widetilde{C}} - \left(\boldsymbol{M}\dot{\hat{\mathbb{J}}}^{\dagger} + \boldsymbol{C}\hat{\mathbb{J}}^{\dagger}\right)\dot{\boldsymbol{s}}_r - \boldsymbol{M}\hat{\mathbb{J}}^{\dagger}\ddot{\boldsymbol{s}}_r - \frac{\partial \mathscr{V}}{\partial \boldsymbol{q}_{\widetilde{C}}}\right\} + \widetilde{\boldsymbol{\theta}}_k\boldsymbol{L}_k^{\dashv}\dot{\boldsymbol{\theta}}_k + \boldsymbol{e}_s^{\intercal}\boldsymbol{K}_p\dot{\boldsymbol{e}}_s \tag{5.29}$$

If we apply the feed back control law in Equation (5.17) and the parameter update law in Equation (5.7), the time derivative of the Lyapnov function candidate becomes

$$\begin{aligned}
\frac{\partial V}{\partial t} &= -\left(\hat{\mathbb{J}}\boldsymbol{v}_q\right)^{\intercal}\boldsymbol{K}_p\boldsymbol{e}_s + \left(\boldsymbol{\theta}_k - \hat{\boldsymbol{\theta}}_k\right)^{\intercal}\boldsymbol{Y}_k^{\intercal}\boldsymbol{K}_p\boldsymbol{e}_s + \boldsymbol{e}_s^{\intercal}\boldsymbol{K}_p\dot{\boldsymbol{e}}_s\ , \\
&= -\left(\hat{\mathbb{J}}\boldsymbol{v}_q\right)^{\intercal}\boldsymbol{K}_p\boldsymbol{e}_s - \left(\boldsymbol{Y}_k\boldsymbol{\theta}_k\right)^{\intercal}\boldsymbol{K}_p\boldsymbol{e}_s + \left(\boldsymbol{Y}_k\hat{\boldsymbol{\theta}}_k\right)^{\intercal}\boldsymbol{K}_p\boldsymbol{e}_s + \boldsymbol{e}_s^{\intercal}\boldsymbol{K}_p\dot{\boldsymbol{e}}_s\ , \\
&= -\left(\hat{\mathbb{J}}\boldsymbol{v}_q\right)^{\intercal}\boldsymbol{K}_p\boldsymbol{e}_s - \dot{\boldsymbol{s}}^{\intercal}\boldsymbol{K}_p\boldsymbol{e}_s + \dot{\hat{\boldsymbol{s}}}^{\intercal}\boldsymbol{K}_p\boldsymbol{e}_s + \boldsymbol{e}_s^{\intercal}\boldsymbol{K}_p\dot{\boldsymbol{e}}_s\ .
\end{aligned} \tag{5.30}$$

To deal with the mixed term $\boldsymbol{e}_s^{\intercal}\boldsymbol{K}_p\dot{\boldsymbol{e}}_s$, we apply the signal vector $\boldsymbol{s}_r$ which is previously defined in Equation (5.20). We have

$$\begin{aligned} \boldsymbol{e}_s^\intercal \boldsymbol{K}_p \dot{\boldsymbol{e}}_s &= \boldsymbol{e}_s^\intercal \boldsymbol{K}_p \left(\dot{\boldsymbol{s}} - \dot{\boldsymbol{s}}^*\right) \ , \\ &= \boldsymbol{e}_s^\intercal \boldsymbol{K}_p \left\{ \dot{\boldsymbol{s}} - \left(\dot{\boldsymbol{s}}_r + \alpha \boldsymbol{e}_s\right) \right\} \\ &= \boldsymbol{e}_s^\intercal \boldsymbol{K}_p \dot{\boldsymbol{s}} - \boldsymbol{e}_s^\intercal \boldsymbol{K}_p \dot{\boldsymbol{s}}_r - \boldsymbol{e}_s^\intercal \boldsymbol{K}_p \alpha \boldsymbol{e}_s \ . \end{aligned} \tag{5.31}$$

With Equation (5.21), and by substituting Equation (5.31) into Equation (5.30), we have

$$\frac{\partial V}{\partial t} = -\left(\hat{\mathbb{J}} \boldsymbol{v}_q\right)^\intercal \boldsymbol{K}_p \boldsymbol{e}_s + \boldsymbol{e}_s^\intercal \boldsymbol{K}_p \boldsymbol{v}_s - \boldsymbol{e}_s^\intercal \boldsymbol{K}_p \alpha \boldsymbol{e}_s \ . \tag{5.32}$$

Finally, we apply Equation (5.24) to the time derivative of the Lyapnov function candidate. It becomes

$$\begin{aligned} \frac{\partial V}{\partial t} &= -\boldsymbol{v}_s^\intercal \boldsymbol{K}_p \boldsymbol{e}_s + \boldsymbol{e}_s^\intercal \boldsymbol{K}_p - \boldsymbol{e}_s^\intercal \boldsymbol{K}_p \alpha \boldsymbol{e}_s \ , \\ &= -\boldsymbol{e}_s^\intercal \boldsymbol{K}_p \alpha \boldsymbol{e}_s \ , \\ &\preceq 0 \ . \end{aligned} \tag{5.33}$$

Since V has been proven to be nonincreasing and bounded below, we have $\boldsymbol{v}_q \in \mathcal{L}^\infty$, $\widetilde{\boldsymbol{\theta}}_k \in \mathcal{L}^\infty$, $\boldsymbol{e}_s \in \mathcal{L}^\infty$. In addition, the joint space sliding vector $\boldsymbol{v}_q$ in Equation (5.23) yields the inequality $\|\dot{\boldsymbol{q}}_{\widetilde{C}}\|_{\mathcal{L}^\infty} - \|\dot{\boldsymbol{q}}_r\|_{\mathcal{L}^\infty} \leq \|\dot{\boldsymbol{q}}_{\widetilde{C}} - \dot{\boldsymbol{q}}_r\|_{\mathcal{L}^\infty} = \|\boldsymbol{v}_q\|_{\mathcal{L}^\infty}$. This inequality indicates that $\dot{\boldsymbol{q}}_{\widetilde{C}} \in \mathcal{L}^\infty$. On the other hand, when we refer to Equations (3.22), and (4.11), we find that all elements in kinematic Jacobian matrix of the manipulator with mounting error are bounded since the multiplication of the rotation matrices will still remains bounded and the cross product of two bounded vectors is bounded. For the case where the projection target does not leave the field of view, the terms u_1, v_1, u_2, v_2 in Equation (3.35) are bounded which makes $\boldsymbol{J}_s$ bounded. Therefore, all elements j_{ik} in the matrix $\mathbb{J}$ are bounded, so $\mathbb{J}$ satisfies the definition of matrix infinity norm, where

$$\|\mathbb{J}\|_{\mathcal{L}^\infty} = \max_i \left(\sum_k |j_{ik}| \right) \leq c \leq \infty \ . \tag{5.34}$$

For this reason, the matrix $\mathbb{J}$ can be seen as a bounded operator. Since $\mathbb{J}$ is a bounded operator and $\dot{\boldsymbol{q}}_{\widetilde{C}} \in \mathcal{L}^\infty$, from the equation $\dot{\boldsymbol{s}} = \mathbb{J}\dot{\boldsymbol{q}}_{\widetilde{C}}$ we have $\dot{\boldsymbol{s}} \in \mathcal{L}^\infty$. Given a smooth target tragectory $\dot{\boldsymbol{s}}^* \in \mathcal{L}^\infty$ and using the fact that $\dot{\boldsymbol{s}} \in \mathcal{L}^\infty$, the time derivative of the tracking error satisfies

$$\dot{\boldsymbol{e}}_s = \underbrace{\dot{\boldsymbol{s}}}_{\in \mathcal{L}^\infty} - \underbrace{\dot{\boldsymbol{s}}^*}_{\in \mathcal{L}^\infty} \ . \tag{5.35}$$

We conclude that $\dot{\boldsymbol{e}}_s \in \mathcal{L}^\infty$. Now we have both $\boldsymbol{e}_s, \dot{\boldsymbol{e}}_s \in \mathcal{L}^\infty$ and $\boldsymbol{e}_s \in \mathcal{L}^2$. According to Barbalat's lemma, it must be the case that $\lim_{t\to\infty} \boldsymbol{e}_s(t) \to \mathbf{0}$.

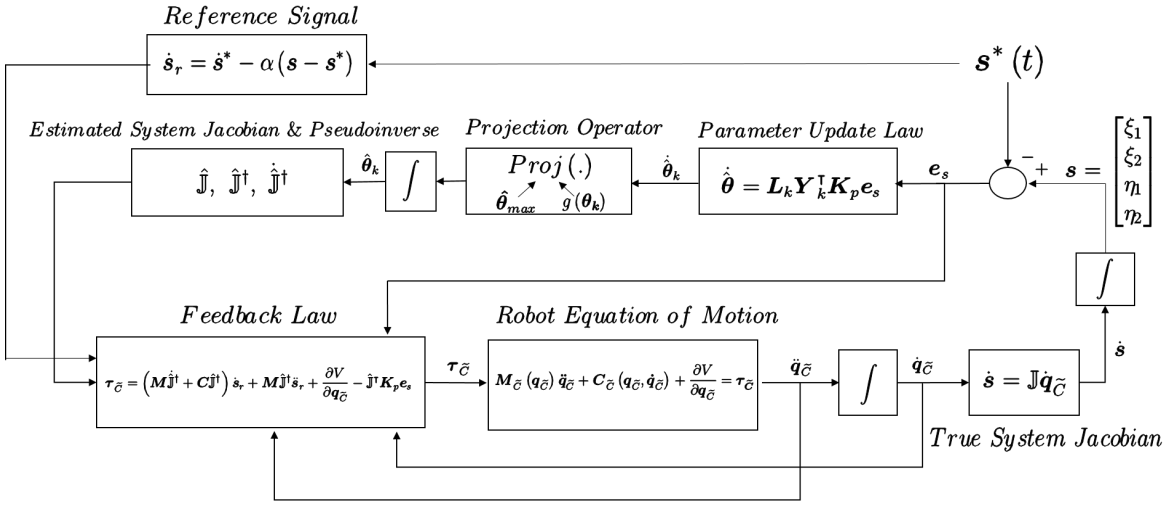


Figure 5.2: The block diagram for implementing the tracking controller

Figure 5.2 depicts a block diagram that represents the tracking controller. However, now the desired image feature location $\boldsymbol{s}^*(t)$ in camera pixel space is time dependent. It is used to form the reference signal. All the terms in the signal are known and measurable. In an application, $\boldsymbol{s}^*(t)$ can be the desired trajectory in pixel space, or it can be the measured in image space from some moving object. The tracking controller contains the blocks *Parameter Update Law*, *Projection Operator*, *Refernce Signal*, *Estimated System Jacobian & Pseudoinverse*, and *Feedback Law*. As in Figure 5.2, the controller generates the computed torque $\boldsymbol{\tau}_{\tilde{C}}$ and applies the torque on the robot links to close the control loop.

In this chapter, the details of the regressor form and two types of the controllers, the calibration controller, and the tracking controller, have been carefully defined. The purpose of the calibration controller is to align the projector-camera unit to the projection target so that the relative pose is the same as the artists defined during rendering. The tracking controller is applied once the robot is well calibrated. The goal of the tracking controller is to achieve target tracking, so that the rendered image coincides with the projected image.

Chapter 6

Simulation

This chapter describes several simulations that have been performed to validate the control law with different combinations of values for the unknown camera parameters. In this thesis, the four calibration cases are presented. We first consider a case in which there is no intrinsic or extrinsic uncertainty. Next we present a case in which there is extrinsic uncertainty only. The third study includes intrinsic uncertainty only. Finally we present a case study in which there is both extrinsic and intrinsic uncertainty. For the cases which include extrinsic parameter uncertainty, we focus on the uncertainty that arises due to the relative orientation of the frames C and $\widetilde{C}$. We assume there there is no uncertainty in the displacement component of the extrinsic camera parameters, so that $\boldsymbol{r}^C_{\widetilde{C},C} = \mathbf{0}$. Specifically, the uncertainty is restricted to the case where misalignment of the camera $\boldsymbol{z}$ axis is determined by the rotation matrix $\boldsymbol{R}^C_{\widetilde{C}}$ as prescribed in Equation (6.1),

$$\boldsymbol{R}^C_{\widetilde{C}} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \ , \tag{6.1}$$

where α is an unknown twist about the x-axis of the end effector frame. Due to the bearings-only measurement provided by the camera, small alignment uncertainty about this axis can cause large errors in positioning. This is why we wish to simulate this class of uncertainty. For simulations we use a nominal focal length $f = 1$ and assume that the pixel dimension terms s_u and s_v are unknown. The simulations are performed using MatLab/SIMULINK. The block diagram representation of the system is shown in Figure 5.1. Therefore, the uncertain parameter vector is

$$\hat{\boldsymbol{\theta}}_k = \hat{\boldsymbol{\theta}}_k \left(s_u, s_v, r_{ij} \left(\alpha \right) \right) \ , \tag{6.2}$$

where r_{ij} is the element in the i^{th} row and the j^{th} column of the rotation matrix $\boldsymbol{R}^C_{\widetilde{C}}$, which

is a function of an unknown twist angle α. The exact from of the estimate parameter vector is given in Equation (5.3).

In the uncertainty case study, all four simulations are performed using the SIMULINK implementation of this block diagram. The initial joint states are $\mathbf{q}_0 = \left[0, \frac{\pi}{2}, 0\right]^T$. The states are driven to the final joint states which are defined as $\mathbf{q}_f = \left[0, \frac{\pi}{2}, \frac{\pi}{4}\right]^T$. The initial camera parameter values are chosen for each of the the individual cases. The range from the camera to the projection target T is 2 meters for all simulation cases. In the simulations, we call the projector-camera unit mounting pose as the "camera extrinsic parameter".

6.1 Known intrinsic and extrinsic camera parameters

The system with no uncertainty is shown in Figure 6.1. Since the projector-camera unit is perfectly mounted on the robot, the twist angle $\alpha = 0$. The pixel error, torque across each joint, and the pixel trajectory for the case with no uncertainty are shown in Figure 6.2.

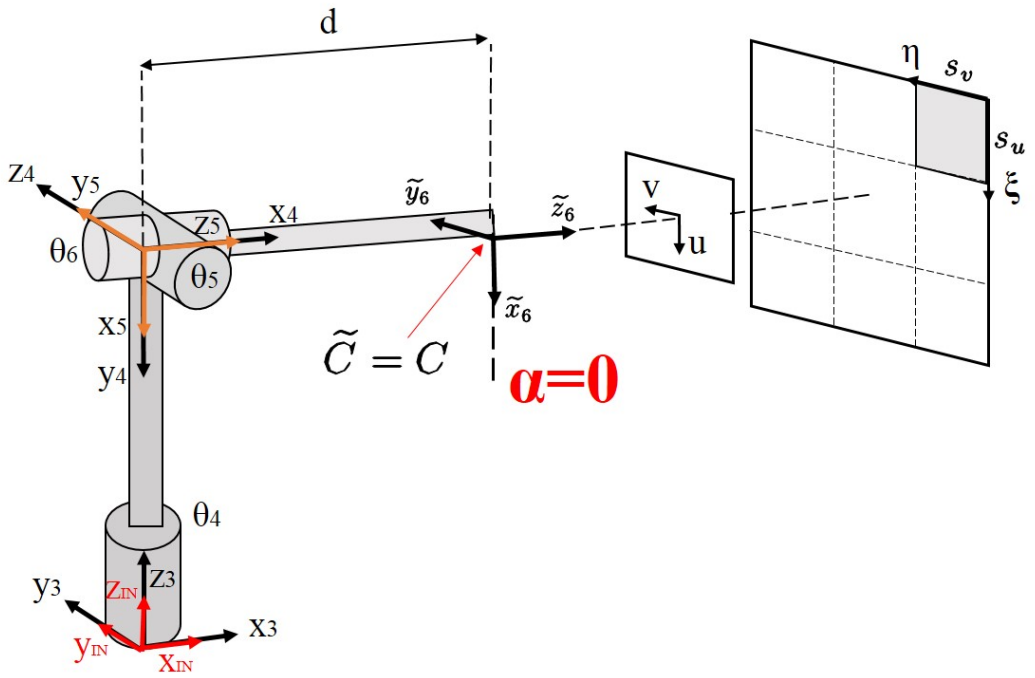
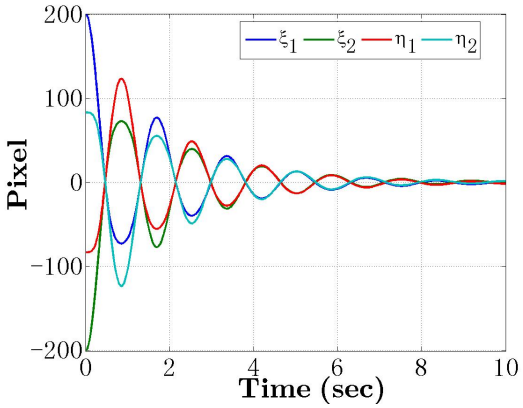
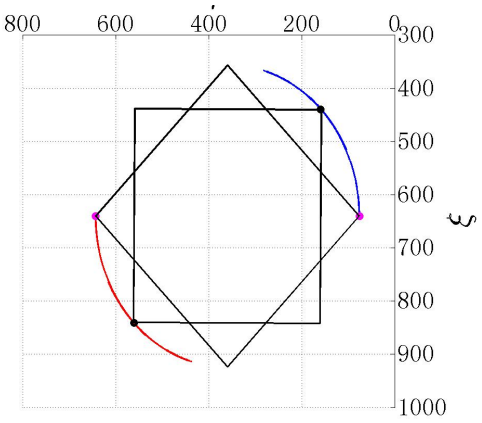


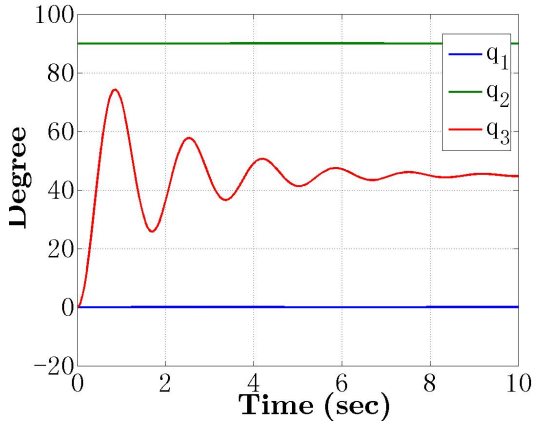
Figure 6.1: No uncertainty occurs in the robot system, the projector-camera unit is ideally mounted.



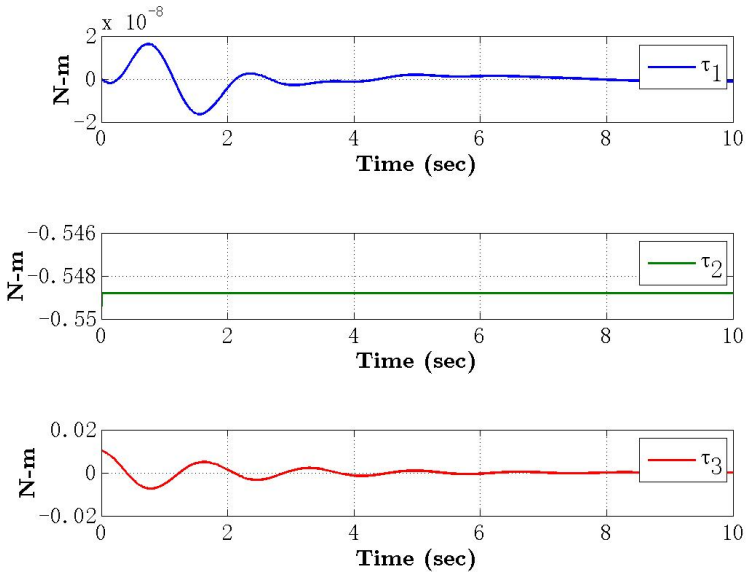
(a) Pixel Error



(b) Pixel space trajectory



(c) Joint angle

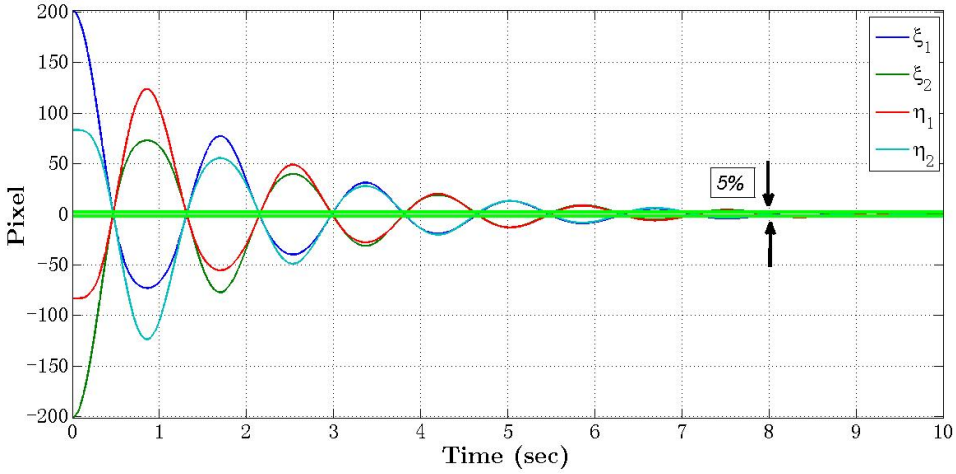


(d) Torque

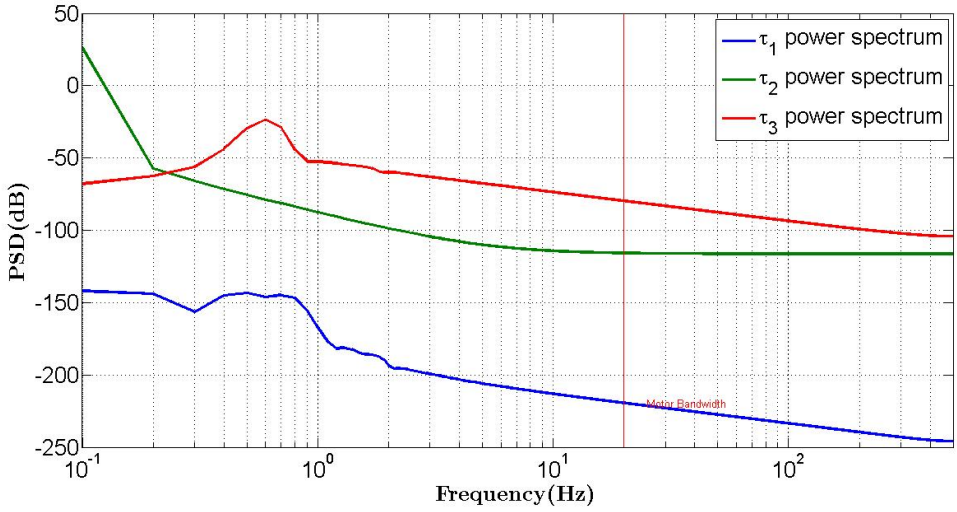
Figure 6.2: Simulation results.

The case with no uncertainty presented above shows that the settling time for the pixel errors are 8 seconds. During this time period, actuation authority requires less than 0.02Nm of torque. According to the trorque transducer sampling frequency, the simulation is ran in $1000Hz$ sampling rate, and the gain matrices $\boldsymbol{K}_p$, $\boldsymbol{K}_v$ are

$$\boldsymbol{K}_p = \begin{bmatrix} 9\times10^{-8} & 0 & 0 & 0 \\ 0 & 9\times10^{-8} & 0 & 0 \\ 0 & 0 & 9\times10^{-8} & 0 \\ 0 & 0 & 0 & 9\times10^{-8} \end{bmatrix}, \boldsymbol{K}_v = \begin{bmatrix} 0.04 & 0 & 0 \\ 0 & 0.04 & 0 \\ 0 & 0 & 0.001 \end{bmatrix}.$$



(a) Pixel error settling time



(b) Torque spectrums

Figure 6.3: Pixel error settling time and torque spectrum of each joint

From the documnet of ABB company [**ABB Motor**], the motor has the bandwidth at $40Hz$ with no load. However, for the case with load, the motor bandwidth frequency will decrease. For this reason, the motor bandwidth frequency with load is set to be $20Hz$. The gain matrices $\boldsymbol{K}_p$ and $\boldsymbol{K}_v$ are tuned so that the system achieve the desired settling time in pixel error and also ensure that within three robot joints, most of the high energy density torque spectrum remain below the moter bandwidth frequency.

6.2 Uncertainty in extrinsic camera parameters

Now we consider the case where the actual system has $\alpha = 2.3^o$ misalignment between real camera pose and ideal camera pose. This misalignment is embedded in the actual robot system and is not known. The controller, it is designed to compenstate for this unknown uncertainty and to achieve the pixel error convergence. Before performing the simulation, it is necessary to assign an initial value of the misalignment angle so that the controller can start updating the estimated misalignment based on this initial value. Therefore, we choose the initial value of the misalignment angle to be $\alpha = 0^o$. In addition, the intrinsic camera parameters $s_u = s_v = 500$, are known in this case.

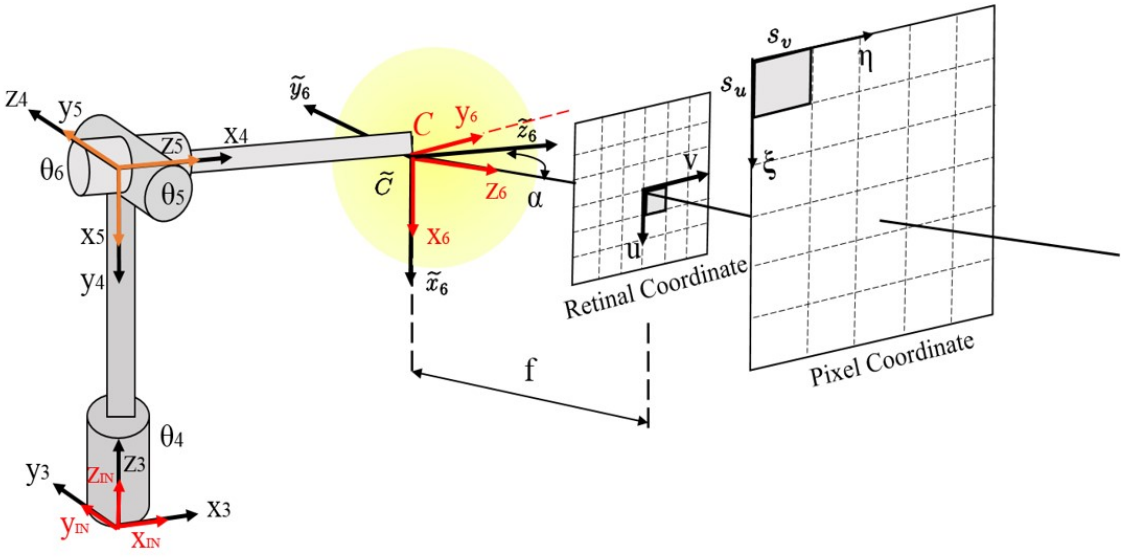


Figure 6.4: Only the projector-camera unit mounting orientation is uncertain *(camera extrinsic parameter)*, but the camera intrinsic parameters are well known

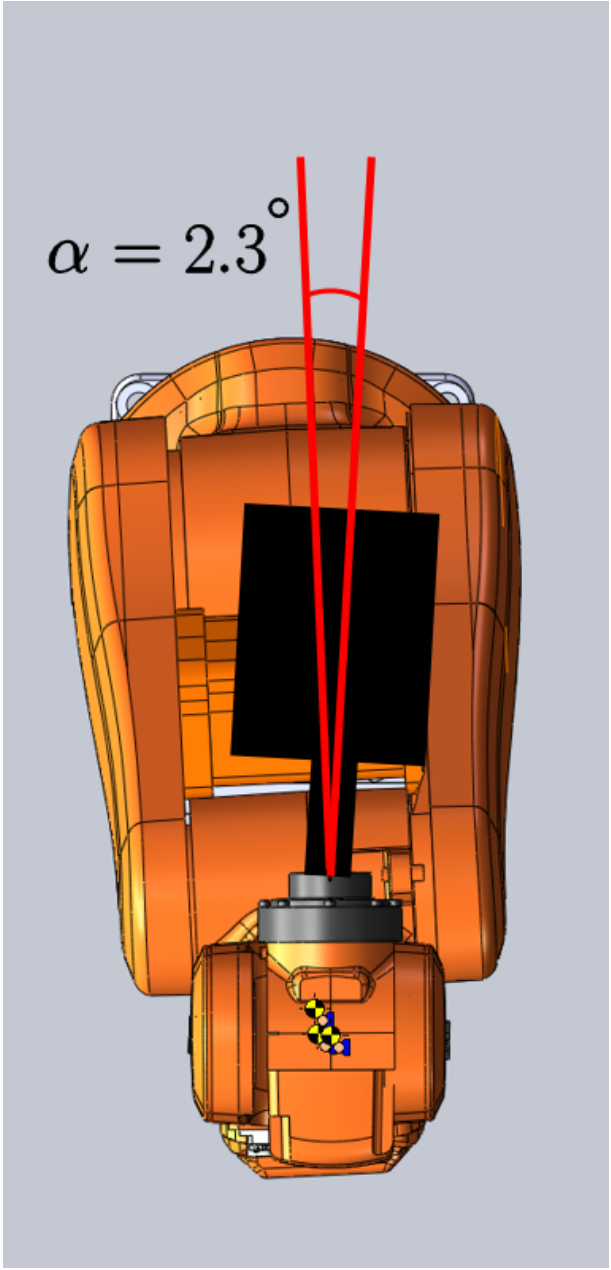
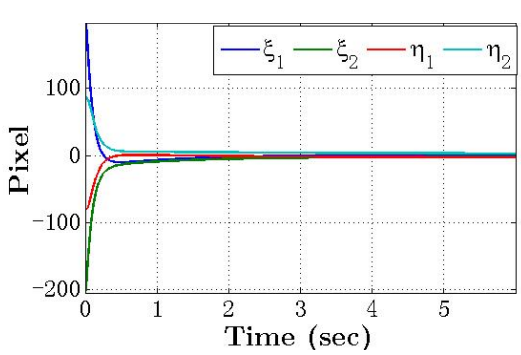
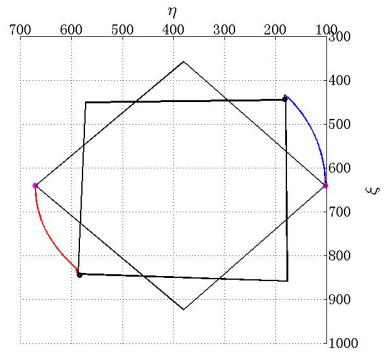


Figure 6.5: Top view of the projection system with uncertain mounting angle α

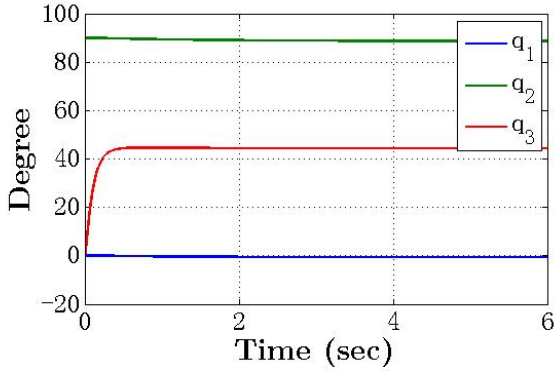
Figure 6.6 depicts the pixel error, actuation torque and the pixel trajectory for the selected feature points.



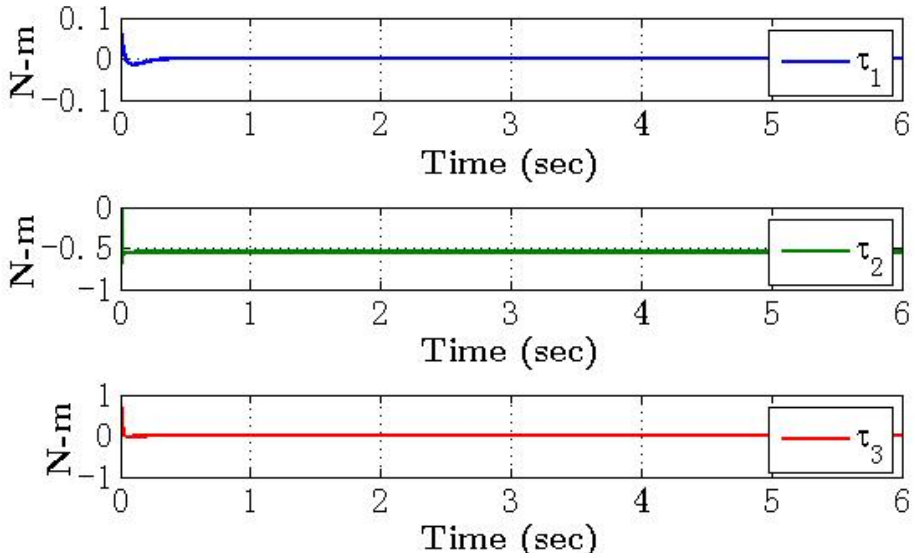
(a) Pixel Error



(b) Pixel space trajectory



(c) Joint angle



(d) Torque

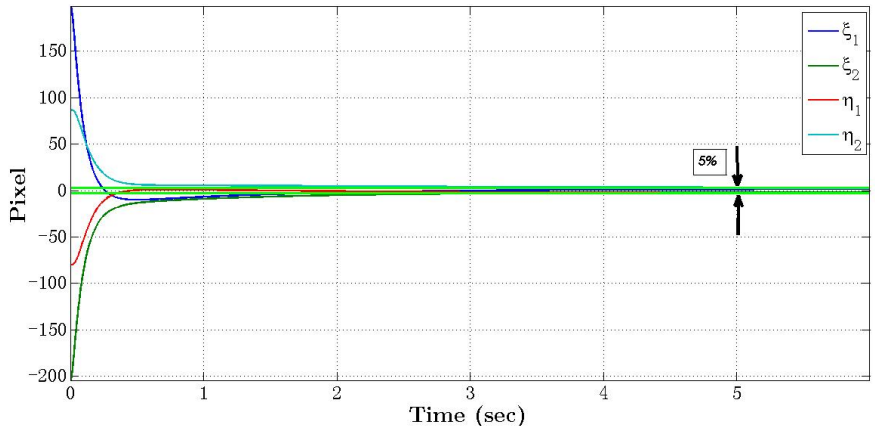
Figure 6.6: Simulation results.

The case with extrinsic uncertainty shows that the settling time for the pixel errors are

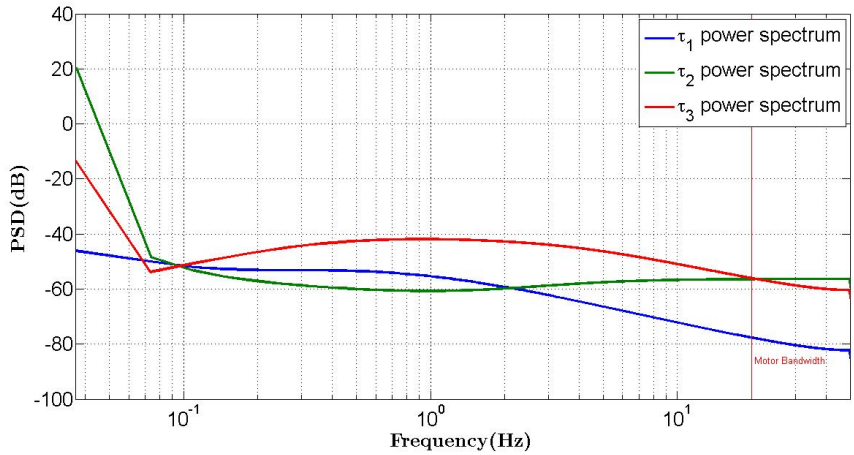
5 seconds. During this time period, actuation authority requires less than 1Nm of torque. According to the trorque transducer sampling frequency, the simulation is ran in $1000Hz$ sampling rate, and the gain matrices $\boldsymbol{K}_p$, $\boldsymbol{K}_v$ are

$$\boldsymbol{K}_p = \begin{bmatrix} 6\times10^{-6} & 0 & 0 & 0 \\ 0 & 6\times10^{-6} & 0 & 0 \\ 0 & 0 & 6\times10^{-6} & 0 \\ 0 & 0 & 0 & 6\times10^{-6} \end{bmatrix}, \boldsymbol{K}_v = \begin{bmatrix} 6 & 0 & 0 \\ 0 & 6 & 0 \\ 0 & 0 & 0.1 \end{bmatrix}.$$

For the case where the system has the extrinsic parameter uncertainty, the gain matrices $\boldsymbol{K}_p$ and $\boldsymbol{K}_v$ are tuned much larger than the case without uncertainty. The higher gain in the controller enable the system converge faster in pixel error. Likewise, the motor is set to have the torque bandwidth at $20Hz$.



(a) Pixel error settling time



(b) Torque spectrums

Figure 6.7: Pixel error settling time and torque spectrum of each joint

6.2.1 Uncertainty in intrinsic camera parameters

In this section, we simulate the system when uncertainty exists only in the intrinsic parameters, shown in Figure 6.8. In the simulation, the actual intrinsic parameters are $s_u = 300$ $s_v = 350$, but the system is initialized with $s_{u,0} = s_{v,0} = 500$. The alignment parameter is $\alpha = 0^o$, which means there is no misalignment between actual camera pose and ideal camera pose. Using these parameter values, the results of the simulation are shown in Figure 6.9,

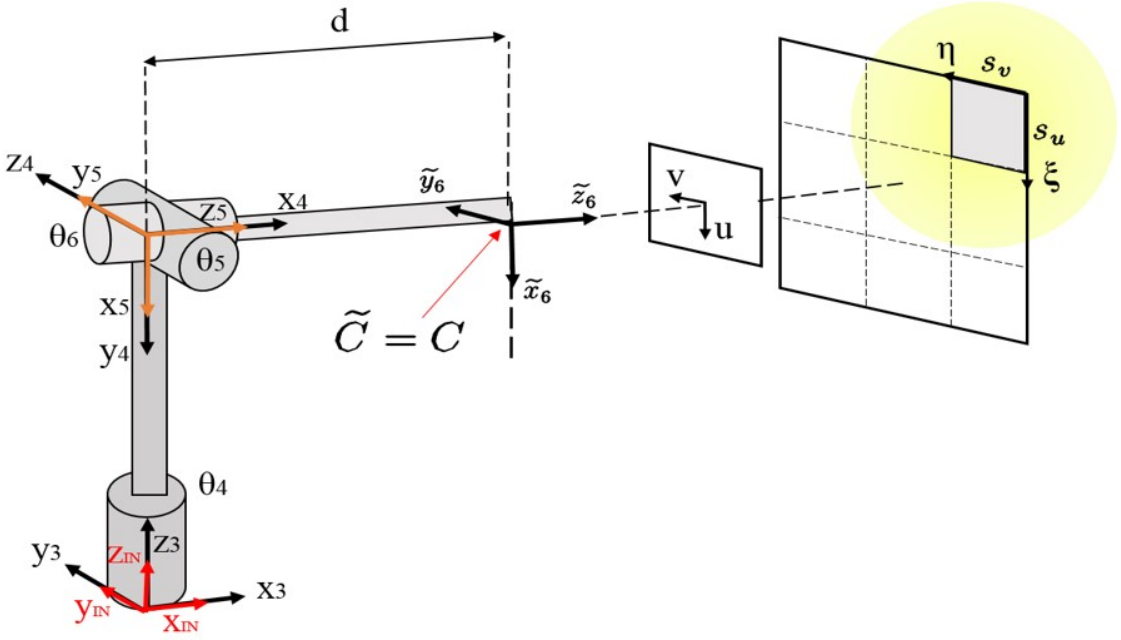
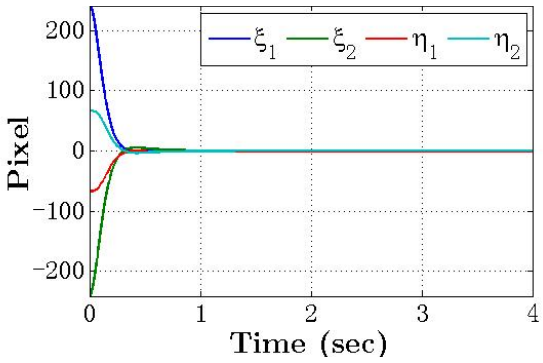
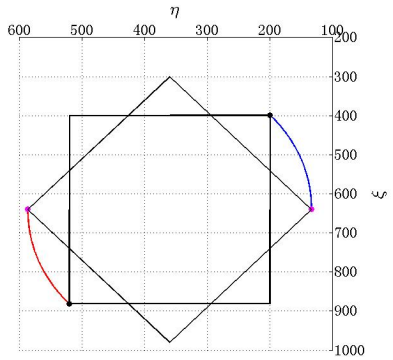


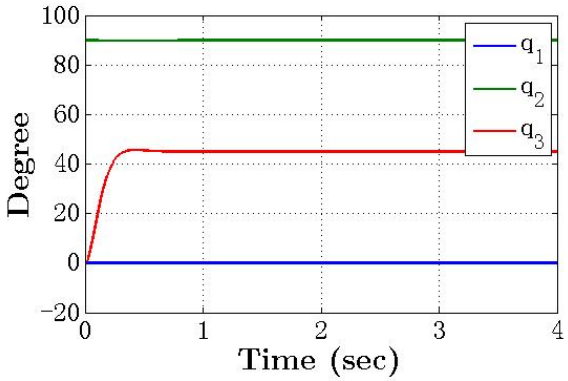
Figure 6.8: Camera intrinsic parameters are uncertain, but the projector-camera unit is ideally mounted



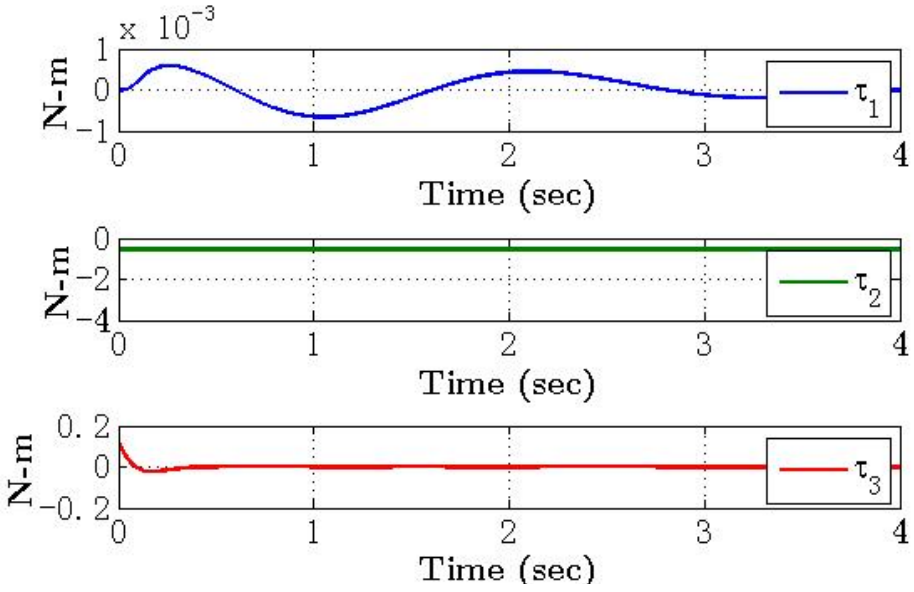
(a) Pixel Error



(b) Pixel space trajectory



(c) Joint angle



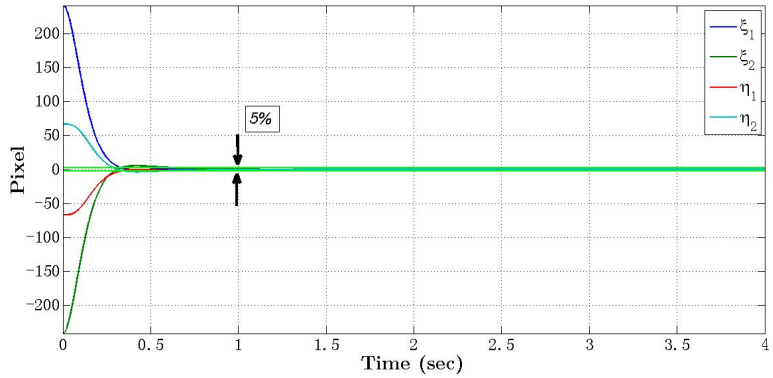
(d) Torque

Figure 6.9: Simulation results.

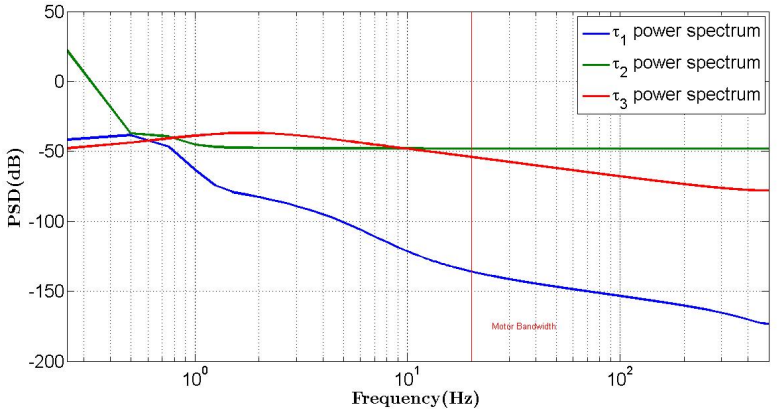
The case with extrinsic uncertainty shows that the settling time for the pixel errors are 1 seconds. During this time period, actuation authority requires less than $0.2Nm$ of torque. According to the trorque transducer sampling frequency, the simulation is ran in $1000Hz$ sampling rate, and the gain matrices $\boldsymbol{K}_p$, $\boldsymbol{K}_v$ are

$$\boldsymbol{K}_p = \begin{bmatrix} 9\times10^{-7} & 0 & 0 & 0 \\ 0 & 9\times10^{-7} & 0 & 0 \\ 0 & 0 & 9\times10^{-7} & 0 \\ 0 & 0 & 0 & 9\times10^{-7} \end{bmatrix}, \boldsymbol{K}_v = \begin{bmatrix} 0.2 & 0 & 0 \\ 0 & 0.2 & 0 \\ 0 & 0 & 0.02 \end{bmatrix}.$$

For the case where the system has only the intrinsic parameter uncertainty, the gain matrices $\boldsymbol{K}_p$ and $\boldsymbol{K}_v$ are tuned larger than the case without uncertainty but lower than the extrinsic uncertain case. Again, the purpose of gain tuning is to achieve desired settling time in pixel error and torque spectrum distribution. Likewise, the motor is set to have the torque bandwidth at $20Hz$.



(a) Pixel error settling time



(b) Torque spectrums

Figure 6.10: Pixel error settling time and torque spectrum of each joint

6.2.2 Uncertainty in extrinsic and intrinsic camera parameters

The final simulation has been conducted with uncertainty in both the intrinsic and the extrinsic camera parameters. The actual system has camera misalignment of $\alpha = 2.3^o$ and the intrinsic parameters are $s_u = 300$ $s_v = 350$. The simulation is initialized with $\alpha_0 = 0^o$ and $s_{u,0} = s_{v,0} = 500$. The derived calibration controller must to compensate for both the camera intrinsic and extrinsic uncertainties at the same time. Simulation results for this case are shown in Figure 6.12

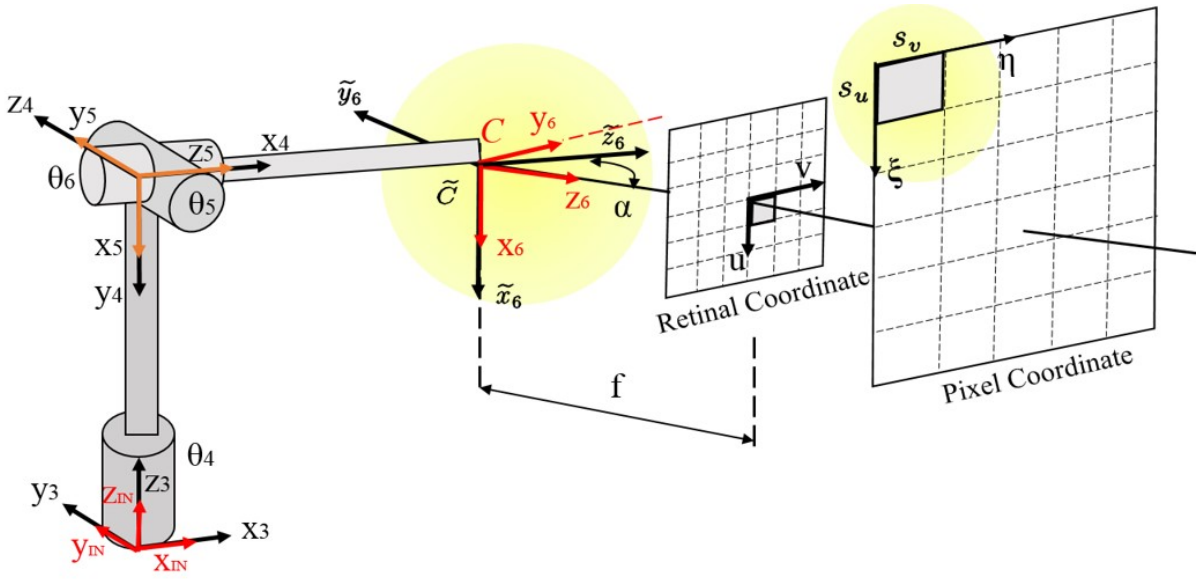
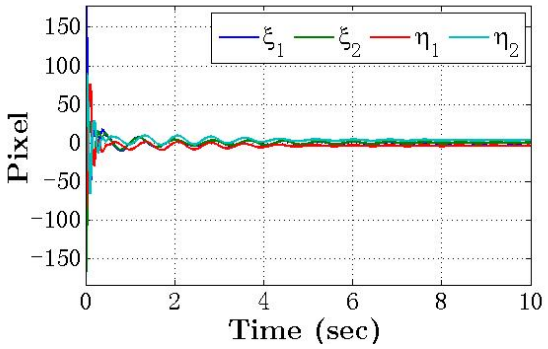
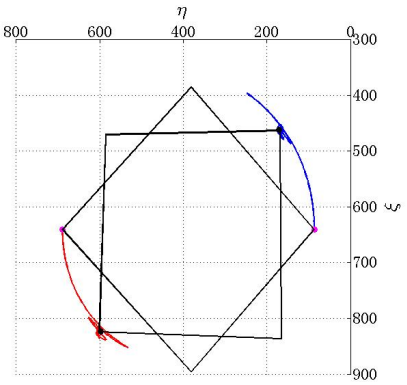


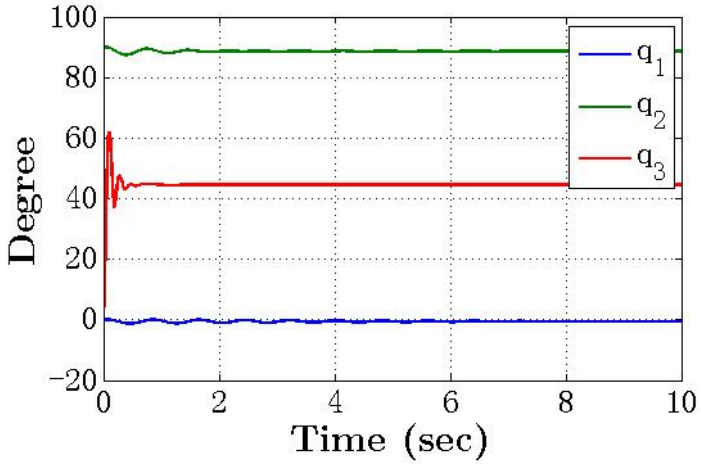
Figure 6.11: Both the projector-camera unit mounting orientation and camera intrinsic parameters are uncertain.



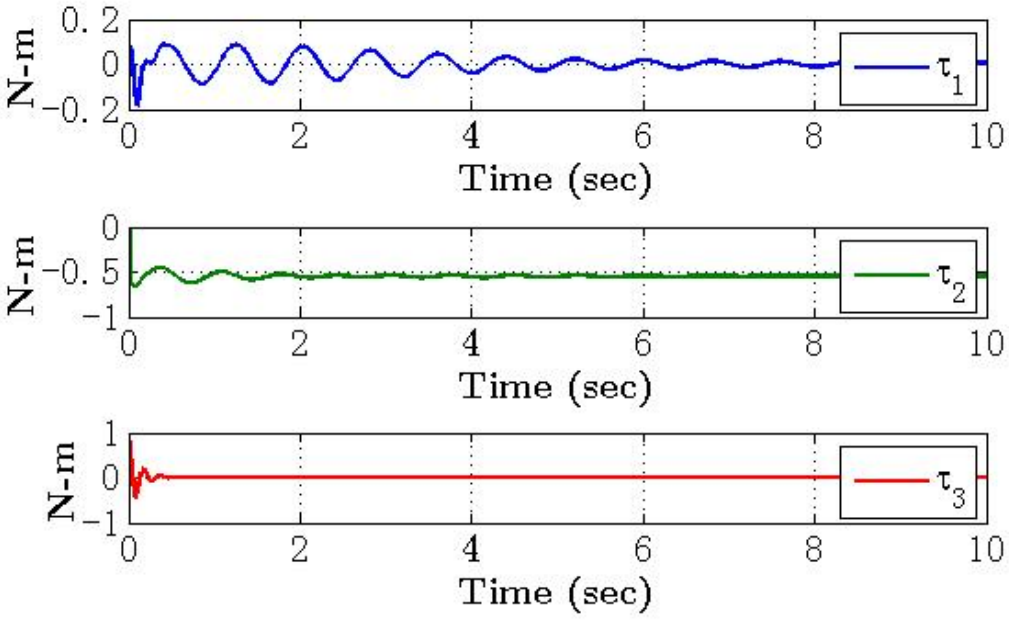
(a) Pixel Error



(b) Pixel space trajectory



(c) Joint angle

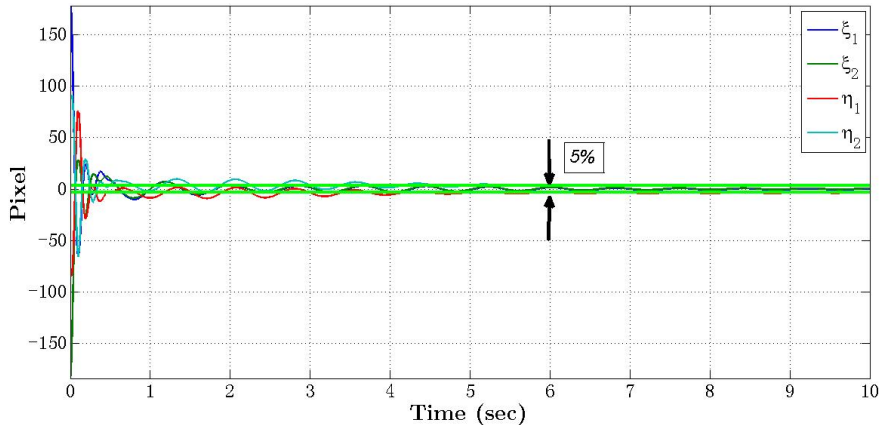


(d) Torque

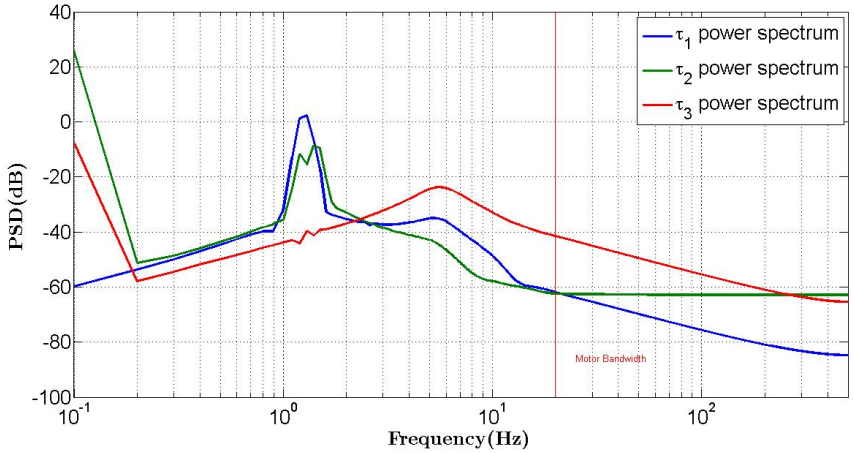
Figure 6.12: Simulation results.

The case with extrinsic uncertainty shows that the settling time for the pixel errors are 6 seconds. During this time period, actuation authority requires less than $1Nm$ of torque. According to the trorque transducer sampling frequency, the simulation is ran in $1000Hz$ sampling rate, and the motor has torque bandwidth frequency at $20Hz$. The gain matrices $\boldsymbol{K}_p$, $\boldsymbol{K}_v$ are

$$\boldsymbol{K}_p = \begin{bmatrix} 8\times10^{-6} & 0 & 0 & 0 \\ 0 & 8\times10^{-6} & 0 & 0 \\ 0 & 0 & 8\times10^{-6} & 0 \\ 0 & 0 & 0 & 8\times10^{-6} \end{bmatrix}, \boldsymbol{K}_v = \begin{bmatrix} 0.085 & 0 & 0 \\ 0 & 0.085 & 0 \\ 0 & 0 & 0.02 \end{bmatrix}.$$



(a) Pixel error settling time



(b) Torque spectrums

Figure 6.13: Pixel error settling time and torque spectrum of each joint

From the gain tuning and Equation (5.6), we find out that the $\boldsymbol{K}_v$ gain can be treated as

damping term in the controller. Since $\boldsymbol{K}_v$ is a 3×3 diagonal matrix, the way to assign more damping on torque number i is to increase the corresponding (i, i) element in $\boldsymbol{K}_v$ diagnal gain matrix. In addition, for the case where all 3 torque trajectories chatter severely, the designer should lower the $\boldsymbol{K}_p$ gain because the system turns out to be too sensitive to the pixel errors.

6.3 Conclusions

In this thesis, we have presented an adaptive control solution for compensating the camera and projector mounting uncertainty and the camera intrinsic uncertainty in the projection system. Different from other static projection system design, this thesis proposes a mobile projection solution that the projector can move along with the projection target. The significant advantage of this design is that the projection area is much wider than the static one. The projected image are rendered by artists, and the system manipulator should calibrate with the projection target before performing target tracking. The purpsoe of the calibration is to maintain the relative pose between the projector and the target same as it is defined in the image rendering software. After the calibration is accomplished, the projection system can begin perform tracking projection.

In the previous chapters, we derive the robot kinematic and dynamic model with Denavit–Hartenberg parameters. In the thesis, we select points as the image feature in pixel space of the camera, however, it can be replaced by other types of image features. When using other types of image feature, the designer will need to derive an interaction matrix that corresponds to the chosen feature. After finishing robot modeling, we introduce the kinematic uncertainty and illustrate the way to write kinematic Jacobian into regressor form. The regressor form can be construct when the object is linear in parameters, and the regression is required for applying the adaptive controller. In addition, we derive the adaptive controller for calibration and target tracking. in the calibration, the desired target points location in pixel coordinate are fixed, but those desired point locations move when we conduct target tracking. In the last chapter, we demonstrate the simulations for the calibration process with different combination of uncertainties in the projection system.

In summury, there are various applications for the vision based tracking system. For the artists, the projection system with mobility will enrich the artistic expression. But in national defense, the laser emitter usually combine with the camera to form a laser tracking system such as the High Energy Laser Mobile Demonstrator *(HEL MD)* developed by Boeing Company. The main goal for this thesis is to study the effect of including the kinematic uncertainties in the image based tracking system. The future work will include simulations of tracking a moving target with kinematic uncertainty and the controller implementation on the robotic hardware described herein.

Bibliography

[1] A.M. Bloch (Author), J. Baillieul (Assistant), P. Crouch (Assistant), Janet Marsden (Assistant). *Nonholonomic Mechanics and Control*. ISBN-13: 978-0387955353.

[2] Andrew J. Kurdila, Alexander Leonessa, Joseph Vignola. *Dynamics and Control of Robotic Systems*. Springer Press.

[3] An-Chyau Huang, Shi-Chang Wu and Wen-Fa Ting. *A FAT-based adaptive controller for robot manipulators without regressor matrix: theory and experiments*. Robotica (2006) volume 24, pp. 205–210.

[4] An-Chyau Huang and Ming-Chih Chien. *Adaptive Control of Robot Manipulators: A Unified Regressor-Free Approach*. World Scientific. ISBN: 978-981-4307-41-3.

[5] A.M. Bloch (Author), J. Baillieul (Assistant), P. Crouch (Assistant), Janet Marsden (Assistant). *Nonholonomic Mechanics and Control (Interdisciplinary Applied Mathematics)*. Springer (September 27, 2007). ISBN-13: 978-0387955353.

[6] Chaumette, François, and Seth Hutchinson. *"Visual servo control. II. Advanced approaches [Tutorial]." Robotics and Automation Magazine*, IEEE 14.1 (2007): 109-118.

[7] C.C.Cheah, C.Liu, and J.J.E.Slotine, *Adaptive Tracking Control for Robots with Unknown Kinematic and Dynamic Properties*, The International Journal of Robotics Research, March 2006 Vol.25 No.3, Pages 283–296.

[8] Cong Wang, Chung-Yen Lin, and Masayoshi Tomizuka *Design of Kinematic Controller for Real-time Vision Guided Robot Manipulators*, 2014 IEEE International Conference on Robotics & Automation (*ICRA*) May 31–June 7, 2014.

[9] C. Copot, C. Lazar,and A. Burlacu *Predictive control of nonlinear visual servoing systems using image moments*, The Institution of Engineering and Technology, Control Theory Appl., 2012, Vol. 6, Iss. 10, pp.1486–1496.

[10] Chien, Ming-Chih and Huang, An-Chyau. *FAT-based Adaptive Visual Servoing for Robots with Varying Uncertainties* . IEEE International Conference on Robotics and Automation Kobe International Conference Center Kobe, Japan, May 12-17, 2009.

[11] Dean C. Karnopp, Donald L. Margolis, Ronald C. Rosenberg. *: System Dynamics: Modeling, Simulation, and Control of Mechatronic Systems, 5th Edition.* ISBN: 978-0-470-88908-4.

[12] E. Malis, F. Chaumette, and S. Boudet. *2-1/2-d visual servoing.* IEEE Trans on Robotics and Automation,15(2):238–250, April 1999.

[13] François Chaumette. *Image Moments: A General and Useful Set of Features for Visual Servoing.* IEEE TRANSACTIONS ON ROBOTICS, VOL. 20, NO. 4, AUGUST 2004.

[14] FranÃ§ois Chaumette *Image Moments: A General and Useful Set of Features for Visual Servoing*, IEEE TRANSACTIONS ON ROBOTICS, VOL. 20, NO. 4, AUGUST 2004.

[15] Hutchinson, Seth, Gregory D. Hager, and Peter I. Corke. *"A tutorial on visual servo control." Robotics and Automation*, IEEE Transactions on 12.5 (1996): 651-670.

[16] Hassan K. Khalil. *Nonlinear Systems (3rd Edition).* Prentice Hall; 3 edition (December 28, 2001). ISBN-13: 978-0130673893.

[17] Hesheng Wang, Yun-Hui Liu, and Weidong Chen *Visual tracking of robots in uncalibrated environments*, Mechatronics Vol 22, Issue 4, Elsevier, June 2012, Pages 390–397 Visual Servoing SI.

[18] Interface, advance force measurement, <http://www.interfaceforce.com/index.php // ?mod=library&show=23>

[19] Katsuhiko Ogata *State space analysis of control systems (Instrumentation and controls series).* Prentice-Hall; 1st Ed. edition (1967). ASIN: B0006BQIYU.

[20] Liao, Chwen Kai, Metthew J. Bender, Andrew Kurdila, and Steve Southward. "Adaptive Control of a Camera–Projection System using Vision-Based Feedback." In AIAA Modeling and Simulation Technologies Conference, p. 1439. 2016.

[21] Lorenzo Sciavicco and Bruno Siciliano. *Modelling and Control of Robot Manipulators (Advanced Textbooks in Control and Signal Processing).* ISBN-13: 978-1846286414. Springer; 1st ed. 2009 edition.

[22] Lorenzo Sciavicco, and Bruno Siciliano, *Modelling and Control of Robot Manipulators*, Springer, 2000.

[23] Mark W. Spong, Seth Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*, Wiley & Sons, 2006.

[24] Matt J.Bender, Hunter G. McClelland, Gerardo Bledt, Andrew Kurdila, Tomonari Furukawa and Rolf Mueller *Trajectory Estimation of Bat Flight Using a Multi-View Camera System*, AIAA SciTech Modeling and Simulation Technologies Conference , 5–9 January 2015.

[25] Mohammad Keshmiri, and Wen Fang Xie *Augmented Imaged Based Visual Servoing Controller for a 6 DOF Manipulator Using Acceleration Command*, 51st IEEE Conference on Decision and Control December 10–13, 2012.

[26] Peter Corke. *Robotics, Vision and Control: Fundamental Algorithms in MATLAB.* ISBN-13: 978-3642201431.

[27] Petros Ioannou, and Jing Sun, *Robust Adaptive Control*, Dover Publications, 2012.

[28] Richard Hartley, Andrew Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press; 2 edition. ISBN-13: 978-0521540513.

[29] Reza N. Jazar. *Advanced Dynamics: Rigid Body, Multibody, and Aerospace Applications.*Wiley; 1 edition (March 29, 2011). ISBN-13: 978-0470398357.

[30] Robert L. Williams II, Douglas A. Lawrence. *Linear State-Space Control Systems.* Wiley; 1 edition (February 9, 2007). ISBN-13: 978-0471735557.

[31] S.S. Mehta, and T.F. Burks, *Vision-based control of robotic manipulator for citrus harvesting,*Computers and Electronics in Agriculture Vol 102, Elsevier, March 2014, Pages 146–158,2014.

[32] ABB Company. *Technical guide-Direct torque control -the world's most advanced AC drive technology,*

[33] Veit Elser, University of California Irvine. Course P3318: Analytical Mechanics.

[34] Yi Ma, Stefano Soatto, Jana Kosecka , S. Shankar Sastry. *An Invitation to 3-D Vision: From Images to Geometric Models.* Springer; 1St Edition edition. ISBN 978-0-387-21779-6