creative commons
COMMONS DEED

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

Model Predictive Visual Guidance for Shortest Distance

A Dissertation

by

HOSEONG SEO

Presented to the Faculty of the Graduate School of
Seoul National University
in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE

School of Mechanical & Aerospace Engineering

Seoul National University
Supervisor : Professor H. Jin Kim

December 2015

to my

MOTHER, FATHER, and BROTHER

with love

Abstract

Model Predictive Visual Guidance for Shortest Distance

Hoseong, Seo
Department of Mechanical & Aerospace Engineering
The Graduate School
Seoul National University

The objective of vision-based guidance is to construct guidance command which steers the camera to the desired pose using visual measurement. This thesis concentrates on design of a guidance command which the trajectory of camera could be as straight as possible, based on a model predictive approach. A straight guidance trajectory is intended to prevent unnecessary movement of the camera. Model predictive approach is utilized due to its capability to handle system constraints, and its robustness with respect to scale uncertainty. Overall process of this thesis is 1) to generate appropriate reference feature trajectory, and 2) to construct a guidance command which follows the reference feature trajectory using an optimization technique. To generate time-independent reference feature trajectory which leads to straight guidance trajectory, partial pose estimation is performed. Estimated pose is used for feature parameterization and definition of the local reference feature is proposed. To track the reference feature trajectory, an additional term is formulated in the cost function to be optimized. Given the assumption of invariant feature depth during the prediction process, computational complexity of optimization is highly reduced, which makes the proposed algorithm more applicable to real environments. From simulation, strength of the proposed algorithm is validated. Experimental results demonstrate more direct guidance trajectory is achieved with reasonable computation time.

Keyword : Vision-based guidance, Visual predictive control (VPC), Model prediction, Optimization

Student Number : 2014-20672

Table of Contents

Page

Abstract . . . iii

Table of Contents . . . iv

List of Figures . . . v

List of Tables . . . vi

Nomenclature . . . vii

Chapter

1 Introduction . . . 1

1.1 Related works . . . 3

1.2 Thesis contribution . . . 5

1.3 Thesis outline . . . 5

2 Problem Formulation . . . 6

3 Reference Feature Generation . . . 8

3.1 Partial pose estimation . . . 8

3.2 Feature parameterization . . . 11

4 VPC Configuration . . . 15

4.1 Prediction model . . . 15

4.2 Cost formulation . . . 17

4.3 Optimization setup . . . 18

5 Simulations . . . 20

5.1 Simulation settings . . . 20

5.2 Simulation results . . . 22

6 Experiments . . . 24

6.1 Experiment settings . . . 24

6.2 Experiment results . . . 28

7 Conclusion . . . 32

References . . . 34

List of Figures

1.1 Illustration of the previous works 4

2.1 Coordinate system of the desired and the initial camera frames 7

3.1 Process of partial pose estimation 9
3.2 Illustration of the linearly interpolated partial poses 12
3.3 Graphical representation of the several features in normalized image plane 14

4.1 Graphical representation of the prediction model in VPC 16

5.1 Environmental setup for the simulations 21
5.2 Comparative results of the simulations 23

6.1 Experimental setup 25
6.2 Initial and desired configuration of the manipulator 26
6.3 Monte-Carlo result of the optimization calculation time 27
6.4 Experiment results with respect to time 29
6.5 Comparative results of the experiments 30
6.6 Comparision of the VPC calculation time 31

List of Tables

4.1 Organization of the cost terms and constraints in the proposed VPC 19

5.1 Parameter settings in the simulations 21
5.2 Comparison of camera trajectory length in the simulations 22

6.1 Parameter settings in the experiments 26
6.2 Comparison of camera trajectory length in the experiments 31

Nomenclature

v_i	ith component of vector $\mathbf{v}$
$[M]_j$	jth column of matrix M
$\tilde{\mathbf{v}}$	homogenous representation of vector $\mathbf{v}$
$\mathbf{t} = [t_x \quad t_y \quad t_z]^\top$	translation vector
$\mathbf{q} = [q_1 \quad q_2 \quad q_3 \quad q_4]^\top$	unit quaternion
$\mathbf{r} = [\phi \quad \theta \quad \psi]^\top$	ZYX Euler angles
$\xi = [\mathbf{t}^\top \quad \mathbf{r}^\top]^\top$	motion vector from the desired pose to the current pose
$\mathbf{P} \in \mathbb{R}^{3\times1}$	coordinate of the one visual target in camera frame
$\mathbf{p} \in \mathbb{R}^{2\times1}$	coordinate of the projected one visual target in image plane
$\mathbf{s} \in \mathbb{R}^{8\times1}$	feature extract from image
$\mathbf{u} \in \mathbb{R}^{6\times1}$	guidance command, i.e. camera velocity
$a \in [0, 1]$	parameter for the reference feature trajectory
N_p	prediction horizon
dt_p	prediction time step
$g(\cdot) : \mathbb{R}^{6\times1} \to \mathbb{R}^{4\times4}$	mapping from motion vector to homogenous transformation matrix
$R(\cdot) : \mathbb{R}^{4\times1} \to \mathbb{R}^{3\times3} \in SO(3)$	mapping from unit quaternion to rotation matrix
$R_z(\psi) : \mathbb{R} \to \mathbb{R}^{3\times3} \in SO(3)$	rotation matrix about z axis with ψ radian
$R_y(\theta) : \mathbb{R} \to \mathbb{R}^{3\times3} \in SO(3)$	rotation matrix about y axis with θ radian
$R_x(\phi) : \mathbb{R} \to \mathbb{R}^{3\times3} \in SO(3)$	rotation matrix about x axis with ϕ radian
$\pi(\cdot) : \mathbb{R}^{3\times n} \to \mathbb{R}^{2\times n}$	perspective projection of the points in 3D Cartesian space
$\mathbf{vec}(\cdot) : \mathbb{R}^{n\times m} \to \mathbb{R}^{nm\times1}$	concatenation of column vectors in 2D matrix

1 Introduction

Guiding from visual feedback, also known as visual servoing, is widely used for many robotic and aerial systems because of its usefulness in unstructured environments. Goal of visual servoing task is to calculate the velocity command of system so that the set of current visual measurements be identical to the set of desired visual measurements. To guide the system successfully, target objects must be inside of the camera's field of view (FOV) and the calculated guidance command should be admissible in respect to system constraints.

Conventionally, visual servo techniques can be classified into two categories: image-based visual servoing (IBVS) and position-based visual servoing (PBVS) [1]. In the former approach, velocity command is directly formed in image space. Advantages of IBVS are robustness to feature depth uncertainty and easiness in handling visibility constraints. However, IBVS could produce undesirable velocity in cartesian space, and only local stability is guaranteed. The latter approach computes velocity command after estimating the pose of camera using image measurements. PBVS could provide a desirable trajectory in cartesian space, while risking departure of the visual target from camera's FOV. In this circumstance, hybrid visual servoing is introduced to employ the advantages of both approaches [2, 3]. This approach decouples translation and rotation components of the velocity command, and each component of the command is formed in image-space and cartesian-space, respectively.

Some advanced visual-servo techniques were developed from the last decades for improving performance of the visual guidance. In some of works, moment-based image feature was developed [4, 5]. The image moment is computed from contour of the target image, using Green's theorem. By using the image moment as a visual feature, exponential convergence of the feature error is guaranteed. These works were extended to the photometric visual servoing [6]. This work proposed the moment feature from the entire image, not extracting the contour. In other works, mutual information and entropy of the captured image was analyzed and developed for visual servoing [7, 8]. Main advantage of the entropy-based visual servoing is that image processing for feature extraction is unnecessary. So, any image could be used for this visual guidance, but its range of convergence is relatively small. There were other works which utilized the epipolar geometry between the initial and desired image [9], and the trifocal tensor from the initial, desired, and current views [10, 11]. However, most of the aforementioned approaches are not considering several constraints, so they have some limit in applying the visual guidance to real environments.

To consider constraints in the visual guidance, some of works combined visual-servo with path planning. In this group of works, the image feature trajectories were calculated from optimization of homogenous forms [12, 13], potential field method [14], or sampling-based path planner [15, 16] satisfying visibility constraints. The main strategy of these approaches is generation of the image-space feature trajectory, and then calculate a velocity command such that the current feature follows the produced trajectory. But, these approaches require global 3D information of the environment for feasible path planning. Moreover, the planned path is incapable of reflecting dynamic alteration of the environments, so the image space path must be re-calculated whenever surrounding changes.

Alternatively, an admissible velocity command could be computed from the model predictive control (MPC) approach [17, 18]. Several works in this category, also known as visual predictive control (VPC), movement of the image feature was predicted from the current system states and inputs. The optimal velocity command could be computed by solving constrained nonlinear optimization. Compared to path planning based method, MPC based method is capable of handling time-variant surroundings and scale uncertainty, because the optimization must be solved at every control or guidance step. However, computation of the optimal control action could be highly time-consuming.

1.1 Related works

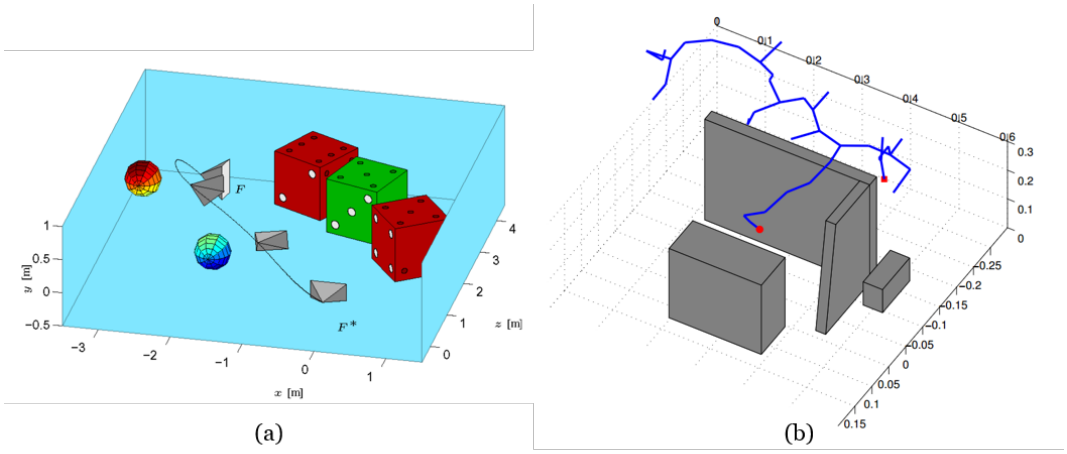
The idea of the prediction model based visual guidance was initially proposed by M. Sauvee, *et al* [17]. They proposed a nonlinear MPC formulation considering camera projection model, dynamic model of 6 d.o.f manipulator, joint and torque limit of each joint, and visibility constraints. Also, robustness of the guidance command to system modelling error and camera calibration error are validated through some simulations.

G. Allibert, *et al* [18] presented definition of local and global image prediction model, and analyzed characteristic of the each prediction model. Also, effect of the prediction horizon on the convergence behavior was discussed. Simulation results of this work showed that the global model has lager convergence domain than the linear model. Moreover, they validated increasing of the prediction horizon guarantees continuous feature trajectory.

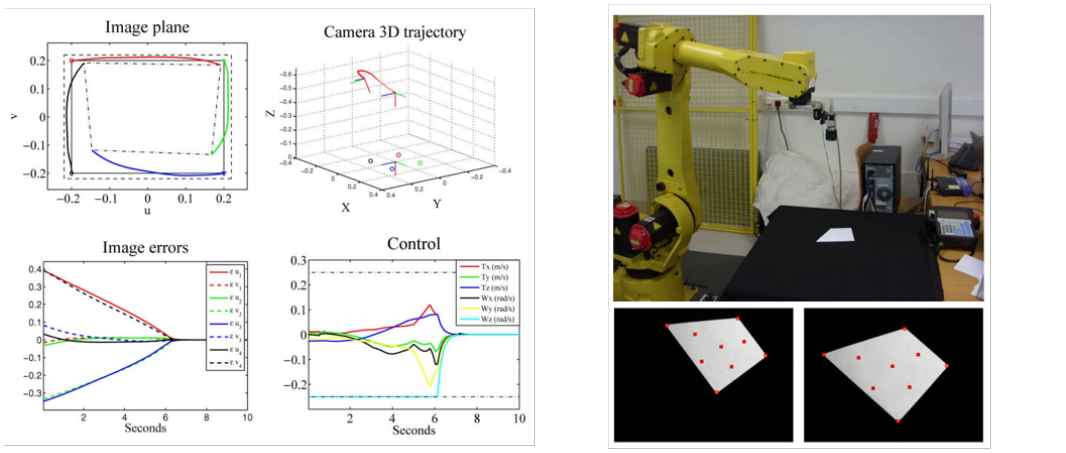
A. Assa, *et al* [19] considered robustness issue of the VPC. Uncertainty model propagated from the image noise was developed and integrated into the optimization. They validated the proposed approach could satisfy the visibility constraint, although there exist uncertainties in image measurement. Also, A. Assa, *et al* [20] proposed the VPC formulation fused with a relative pose error. They assumed relative pose between the current camera frame and the desired camera frame is known, and the error was defined in both image and Cartesian space. The simulation results showed that the VPC considering relative pose could improve stability of the VPC.

A. Burlacu, *et al* [21, 22] performed real-time experiment of VPC on the 6 d.o.f manipulator arm. In this work, the image moment feature was used, because computation of the image moment is more stable and robust than feature point matching algorithm. Also, prediction time step in the prediction model was determined considering computation time of optimization loop and control loop.

D. Lee. *et al* [23] presented vision-based obstacle avoidance in a dynamic environment for a fixed-wing unmanned aerial vehicle. To maintain visibility of the target, visibility constraint was integrated in the nonlinear model predictive control framework (NMPC). They demonstrated the capability of NMPC in respect to collision avoidance with moving target, through camera-in-the-loop simulations.

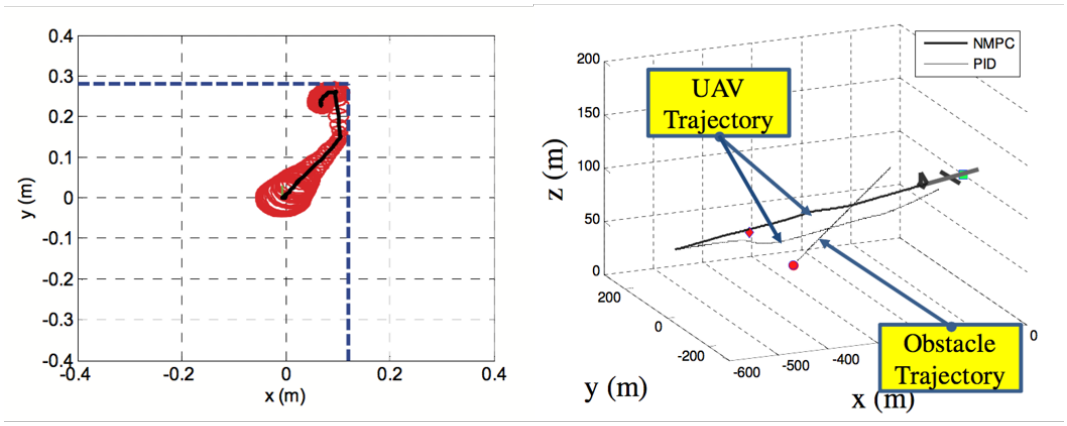


(a) (b)



(c)

(d)



(e) (f)

Figure 1.1: Illustration of the previous works.
(a) G. Chesi *et al* [13]. (b) M. Kazemi *et al* [15]. (c) G. Allibert *et al* [18].
(d) A. Burlacu *et al* [22]. (e) A. Assa *et al* [19]. (f) D. Lee *et al* [23].

1.2 Thesis contribution

The contributions of this thesis are the following: first, we generate time-independent reference feature trajectory for pursing guidance trajectory to be geodesic. Without metric information of the visual target, time-dependent reference feature trajectory is useless, although it could be generated. To construct the reference feature trajectory, partial pose estimation between the initial and the desired image is utilized. After that, we develop additional term of the cost function in VPC for tracking the generated reference feature trajectory. The cost function and its Jacobian are derived and used for the optimization. Finally, advantages of the proposed approach is validated through simulations and experiments. By combining VPC with the reference feature trajectory, we could produce more desirable guidance trajectory in cartesian space without any geometric information of the visual target and much of additional computation effort.

1.3 Thesis outline

The rest of this paper is organized as follows. Chapter 2 introduces the problem formulation. In Chapter 3, methodology for constructing the parameterized reference feature trajectory and definition of the local reference feature are described. In Chapter 4, formulation of proposed VPC is discussed in detail. Simulation and experimental results are presented in Chapter 5 and Chapter 6, respectively. Concluding remarks and potential future works are included in Chapter 7.

2

Problem Formulation

The problem to be solved in this thesis is construct guidance command such that steers the camera from the initial frame to the desired frame:

1) minimizing the length of guidance trajectory.

2) satisfying system constraints.

without metric information of the visual target.

Coordinate system used in this thesis is illustrated in Fig 2.1. We use four corners of a planar target as a visual target. We assume the visual target is planar, and this target is parallel to the desired camera frame. The desired frame and the initial frame are detonated as F_d and F_i respectively. In each camera frame, viewing direction is correspond to z-axis of the frame, and perspective camera model is used for projection of the visual target. Each corner of the target is perspectively projected to a normalized image plane of the camera, and its coordinate is denoted as (x, y). In this figure, grey rectangle is graphical representation of the normalized image plane. We use stack of four corner positions in the image plane as a visual feature.

$$\mathbf{s} = \begin{bmatrix} \mathbf{p}_1^\top & \mathbf{p}_2^\top & \mathbf{p}_3^\top & \mathbf{p}_4^\top \end{bmatrix}^\top = \begin{bmatrix} x_1 & y_1 & x_2 & y_2 & x_3 & y_3 & x_4 & y_4 \end{bmatrix}^\top \tag{2.1}$$

$\mathbf{s}_d$ and $\mathbf{s}_i$ represent visual feature in the desired and the initial camera frames. Also, distance between center of the desired frame and the visual target is assumed to be an arbitrary constant

l. So, coordinate of the visual target represented in the desired camera frame is determined as follow.

$$P_d = \begin{bmatrix} \mathbf{P}_{1,d} & \mathbf{P}_{2,d} & \mathbf{P}_{3,d} & \mathbf{P}_{4,d} \end{bmatrix} = l \begin{bmatrix} x_{1,d} & x_{2,d} & x_{3,d} & x_{4,d} \\ y_{1,d} & y_{2,d} & y_{3,d} & y_{4,d} \\ 1 & 1 & 1 & 1 \end{bmatrix} \tag{2.2}$$

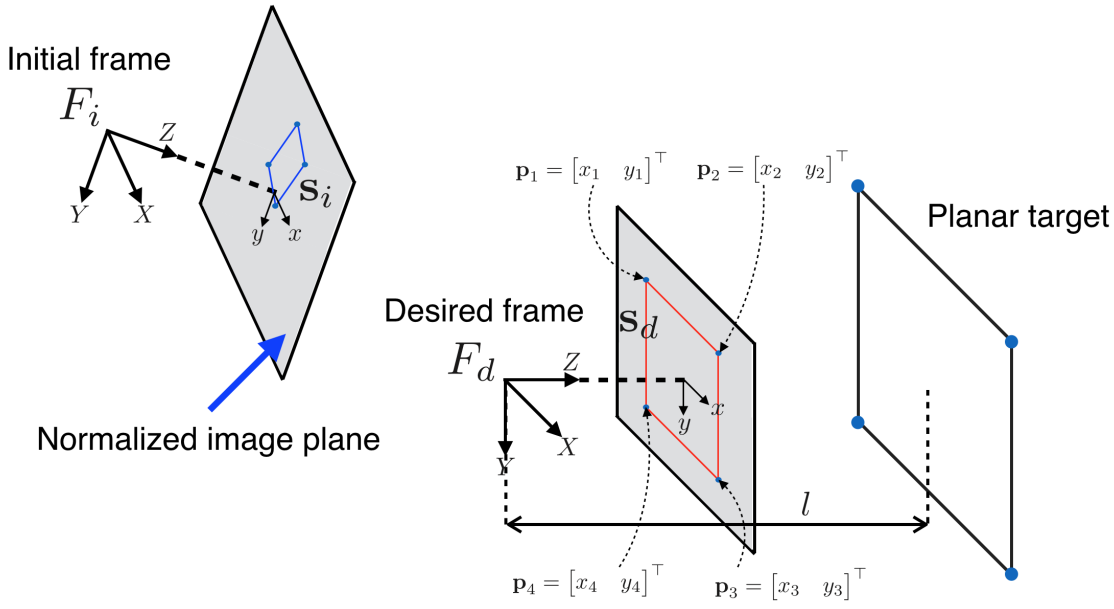


Figure 2.1: Coordinate system of the desired and the initial camera frames

3

Reference Feature Generation

This chapter is organized as follow; Section 3.1 explains the procedure of estimating partial pose between the initial frame and the desired frame. The estimated pose is used in Section 3.2 to parameterize reference feature trajectory with respect to a scalar variable a. Graphical representation of the reference feature trajectory is also illustrated in Section 3.2.

3.1 Partial pose estimation

To generate the reference feature trajectory, partial pose between the initial camera frame and the desired camera frame is required. Unlike rotational component, it is impossible to determine a scale of translation vector between the initial and the desired pose because metric information of the visual target is unknown. So, only *up-to-scale* translation vector could be estimated.

The objective of partial pose estimation is find relative motion vector ξ^* which minimizes difference between a feature in virtual camera frame and a feature in the initial camera frame. This could be interpreted as minimizing following value.

$$C(\xi) = \frac{1}{2}(\mathbf{s}(\xi) - \mathbf{s}_i)^\top (\mathbf{s}(\xi) - \mathbf{s}_i) \tag{3.1}$$

In (3.1), the feature of virtual camera is denoted as $\mathbf{s}(\xi)$, and it is function of the motion vector only. This characteristic is originated from the assumption of this thesis; the visual target

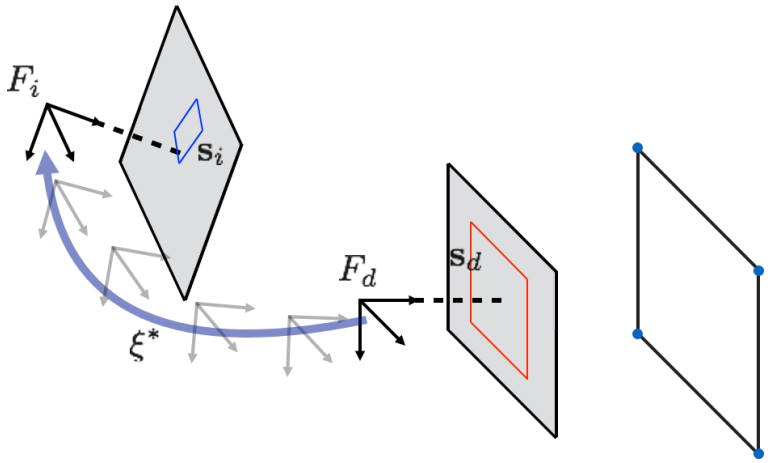


Figure 3.1: Process of partial pose estimation

is parallel to the desired frame and depth of the visual target at the desired frame is an arbitrary constant. Let us say one of the virtual frames as F_v, and its corresponding motion vector is ξ_v. From the definition of motion vector, $g(\xi_v)$ represents homogenous transformation from the desired frame to the virtual frame.

$$g(\xi_v) = \begin{bmatrix} \mathbf{R}_v & \mathbf{t}_v \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \tag{3.2}$$

In (3.2), $\mathbf{R}_v$, $\mathbf{t}_v$ represent a rotation matrix and a *up-to-scale* translation vector from the desired frame to the virtual frame represented in the desired frame. Using (2.2), 3D position of the visual target and corresponding virtual feature could be computed as below.

$$\tilde{P}_v = g(\xi_v)^{-1}\tilde{P}_d \tag{3.3}$$

$$\mathbf{s}_v = \pi(P_v) \tag{3.4}$$

Because (3.1) is only dependent to ξ, we could tackle this problem as an unconstrained minimization of $C(\xi)$. In Fig 3.1, process of optimization is illustrated. The virtual frames are depicted as grey frames in Fig 3.1. Initial guess of the motion vector is null vector. In other words, the initial virtual frame is coincide with the desired frame. Therefore, initial guess of the virtual feature and 3D coordinate of the visual target represented in the virtual frame are equal to corresponding

value determined in the desired frame.

$$\xi_0 = \mathbf{0}_{6\times 1}, \quad \mathbf{s}_0 = \mathbf{s}_d, \quad P_0 = P_d \tag{3.5}$$

Simple gradient-descent update law could be available to this problem. Following equations are iteratively used until norm of the update vector $\delta\xi_k$ is less then some predefined threshold. σ in (3.6) is a positive scalar value, which determines a size of the update law.

$$\delta\xi_k = -\sigma \frac{\partial \mathbf{s}_k}{\partial \xi_k}^{\top} (\mathbf{s}_k - \mathbf{s}_i) \tag{3.6}$$

$$\xi_{k+1} = \xi_k + \delta\xi_k \tag{3.7}$$

$$P_{k+1} = g(\delta\xi_k)^{-1} P_k \tag{3.8}$$

$$\mathbf{s}_{k+1} = \mathbf{vec}(\pi(P_{k+1})) \tag{3.9}$$

In (3.6), Jacobian of the virtual feature with respect to the motion vector is used. In this thesis, we only derive Jacobian of the first feature without loss of generality, because other Jacobian could be derived easily using similar manner. From this part of section, let us use $\mathbf{P} \equiv \mathbf{P}_1$, $\mathbf{p} \equiv \mathbf{p}_1$ for convenience.

From (3.3), 3D coordinate of the visual target represented in the virtual frame and its Jacobian with respect to the motion vector are computed as follow.

$$\mathbf{P}_v = \begin{bmatrix} X_v \\ Y_v \\ Z_v \end{bmatrix} = \mathbf{R}_v^{\top}(\mathbf{P}_d - \mathbf{t}_v) \tag{3.10}$$

$$\frac{\partial \mathbf{P}_v}{\partial \xi} = \begin{bmatrix} \frac{\partial X_v}{\partial \xi} \\ \frac{\partial Y_v}{\partial \xi} \\ \frac{\partial Z_v}{\partial \xi} \end{bmatrix} = \begin{bmatrix} -\mathbf{R}_v^{\top} & | & \frac{d\mathbf{R}_v}{d\phi}^{\top}(\mathbf{P}_d - \mathbf{t}_v) & \frac{d\mathbf{R}_v}{d\theta}^{\top}(\mathbf{P}_d - \mathbf{t}_v) & \frac{d\mathbf{R}_v}{d\psi}^{\top}(\mathbf{P}_d - \mathbf{t}_v) \end{bmatrix} \tag{3.11}$$

Where, derivative of rotation matrices are computed from (3.12).

$$\begin{aligned} \frac{d\mathbf{R}}{d\phi} &= R_z(\psi) R_y(\theta) \left(\frac{d}{d\phi} R_x(\phi) \right) \\ \frac{d\mathbf{R}}{d\theta} &= R_z(\psi) \left(\frac{d}{d\theta} R_y(\theta) \right) R_x(\phi) \\ \frac{d\mathbf{R}}{d\psi} &= \left(\frac{d}{d\psi} R_z(\psi) \right) R_y(\theta) R_x(\phi) \end{aligned} \tag{3.12}$$

Considering perspective projection, position of the projected visual target in the virtual frame and its Jacobian are determined as below.

$$\mathbf{p}_v = \begin{bmatrix} x_v \\ y_v \end{bmatrix} = \begin{bmatrix} X_v/Z_v \\ Y_v/Z_v \end{bmatrix} \tag{3.13}$$

$$\frac{\partial \mathbf{p}_v}{\partial \xi} = \begin{bmatrix} \frac{\partial x_v}{\partial \xi} \\ \frac{\partial y_v}{\partial \xi} \end{bmatrix} = \begin{bmatrix} \frac{1}{Z_v}\frac{\partial X_v}{\partial \xi} - \frac{X_v}{Z_v^2}\frac{\partial Z_v}{\partial \xi} \\ \frac{1}{Z_v}\frac{\partial Y_v}{\partial \xi} - \frac{Y_v}{Z_v^2}\frac{\partial Z_v}{\partial \xi} \end{bmatrix} \tag{3.14}$$

In (3.14), X_v, Y_v, and Z_v are determined from (3.10), and their Jacobian are computed using (3.11). Finally, Jacobian of the feature in the virtual frame with respective to the motion vector is stack of the individual Jacobian represented in (3.14).

$$\frac{\partial \mathbf{s}}{\partial \xi} = \begin{bmatrix} \frac{\partial \mathbf{p}_1}{\partial \xi}^\top & \frac{\partial \mathbf{p}_2}{\partial \xi}^\top & \frac{\partial \mathbf{p}_3}{\partial \xi}^\top & \frac{\partial \mathbf{p}_4}{\partial \xi}^\top \end{bmatrix}^\top \tag{3.15}$$

3.2 Feature parameterization

After partial pose between the initial camera frame and the desired camera frame is estimated, a parameterized reference feature trajectory could be generated. Based on two advantages, we use linear interpolation between the partial pose of the initial frame and the desired frame. The first advantage is, if the reference feature trajectory is generated from linearly interpolated poses, the expected trajectory of the camera is straight line when the reference feature trajectory is appropriately tracked. And the second reason is the linear interpolation needs only one scalar variable; therefore, the reference feature trajectory also be expressed by one variable.

In Fig 3.2, grey frames represent the linearly interpolated poses. Equations related with the linear interpolation is given below.

$$\begin{aligned} \mathbf{t}(a) &= (1-a)\mathbf{t}_i + a\mathbf{t}_d \\ \mathbf{q}(a) &= (1-a)\mathbf{q}_i + a\mathbf{q}_d \end{aligned} \tag{3.16}$$

Where a is an interpolation parameter which ranges in $[0, 1]$, $\mathbf{t}_d$ is a null vector of 3×1, and $\mathbf{q}_d$ is a unit quaternion of zero-rotation. Also, $\mathbf{t}_i$ and $\mathbf{q}_i$ are result of the partial pose estimation, explained in section 3.1. Using (3.16), the first visual target projected to the interpolated pose

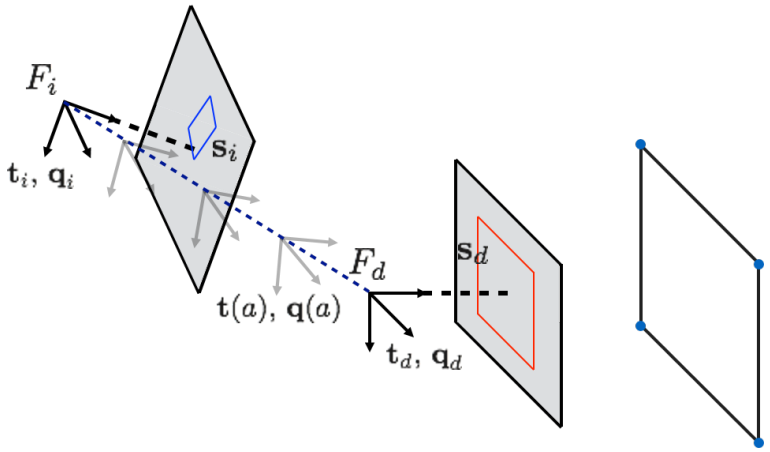


Figure 3.2: Illustration of the linearly interpolated partial poses

could be parameterized with a similar manner of (3.10).

$$\mathbf{P}(a) = \begin{bmatrix} X(a) \\ Y(a) \\ Z(a) \end{bmatrix} = R(\mathbf{q}(a))^\top (\mathbf{P}_d - \mathbf{t}(a)) \tag{3.17}$$

$$\mathbf{p}(a) = \begin{bmatrix} x(a) \\ y(a) \end{bmatrix} = \begin{bmatrix} X(a)/Z(a) \\ Y(a)/Z(a) \end{bmatrix} \tag{3.18}$$

It is natural to calculate derivative of the parameterized feature with respect to a. Moreover, its derivative is required for the following of this thesis. Derivative of (3.16) to (3.18) is computed as below.

$$\frac{d\mathbf{t}(a)}{da} = \mathbf{t}_d - \mathbf{t}_i, \quad \frac{d\mathbf{q}(a)}{da} = \mathbf{q}_d - \mathbf{q}_i \tag{3.19}$$

$$\frac{dR(a)}{da} = \frac{\partial R}{\partial \mathbf{q}} \frac{\partial \mathbf{q}}{\partial a} = \sum_{i=1}^{4} \frac{dR}{dq_i} \frac{dq_i}{da} \tag{3.20}$$

$$\frac{d\mathbf{P}(a)}{da} = \begin{bmatrix} \frac{dX(a)}{da} \\ \frac{dY(a)}{da} \\ \frac{dZ(a)}{da} \end{bmatrix} = \frac{dR(a)}{da}^\top (\mathbf{P}_d - \mathbf{t}(a)) - R(a)^\top \frac{d\mathbf{t}(a)}{da} \tag{3.21}$$

$$\frac{d\mathbf{p}(a)}{da} = \begin{bmatrix} \frac{1}{Z(a)} \frac{dX(a)}{da} - \frac{X(a)}{Z(a)^2} \frac{dZ(a)}{da} \\ \frac{1}{Z(a)} \frac{dY(a)}{da} - \frac{Y(a)}{Z(a)^2} \frac{dZ(a)}{da} \end{bmatrix} \tag{3.22}$$

Similar to (3.15), derivative of the parameterized feature with respect to a is stack of the individual derivatives defined in (3.22)

$$\frac{\partial \mathbf{s}(a)}{\partial a} = \begin{bmatrix} \frac{\partial \mathbf{p}_1(a)}{\partial a}^\top & \frac{\partial \mathbf{p}_2(a)}{\partial a}^\top & \frac{\partial \mathbf{p}_3(a)}{\partial a}^\top & \frac{\partial \mathbf{p}_4(a)}{\partial a}^\top \end{bmatrix}^\top \tag{3.23}$$

To utilize the parameterized reference feature trajectory in the prediction model of VPC, a reference feature correspond to a predicted feature must be calculated. In this context, local reference feature is defined when a new predicted feature $\mathbf{s}_{new}$ is determined.

$$\mathbf{s}_{r,new} = \mathbf{s}(a^*) \tag{3.24}$$

$$a^* = \mathbb{F}(\mathbf{s}_{new}) = argmin \frac{1}{2}(\mathbf{s}(a) - \mathbf{s}_{new})^\top (\mathbf{s}(a) - \mathbf{s}_{new}) \tag{3.25}$$

From (3.24) to (3.25), the local reference feature is one of the reference feature trajectory which is closet to a predicted feature. Calculating a^* requires iterative minimization procedure, so this process is denoted as $\mathbb{F}(\cdot)$. To solve (3.25), gradient-descent update law is available, starting from some initial value.

$$\delta a = -\sigma \frac{d\mathbf{s}(a)}{da}^\top (\mathbf{s}(a) - \mathbf{s}_{new}) \tag{3.26}$$

Note that (3.23) is used to determine update law δa in (3.26).

In the following, variation of a with respect to $\mathbf{s}_{new}$ is used for some calculation. However, it is difficult to calculate $\frac{da}{d\mathbf{s}}$ analytically, because the variable a cannot be explicitly expressed. Therefore, we defined Jacobian of the variable a with respect to the predicted feature as a numerical differentiation.

$$\left(\frac{da}{d\mathbf{s}}\right)_j = \frac{\mathbb{F}(\mathbf{s} + \delta\mathbf{s}_j) - \mathbb{F}(\mathbf{s})}{\epsilon} \tag{3.27}$$

Where ϵ is some arbitrary small value, $\delta\mathbf{s}_j$ represents 8×1 null vector with its j-th element is equal to ϵ.

Fig 3.3 illustrates graphical interpretation of the several features discussed in this chapter. A blue and red rectangle represent image at the initial and the desired camera respectively. The parameterized feature is illustrated as black rectangles and some of them are drawn in the figure. When a new feature $\mathbf{s}_{new}$ is predicted in the prediction model, a green rectangle is selected as the local reference feature derived from (3.24) and (3.25). As presented in Fig 3.3, the green rectangle is one of the black rectangles which is closet to the magenta rectangle.

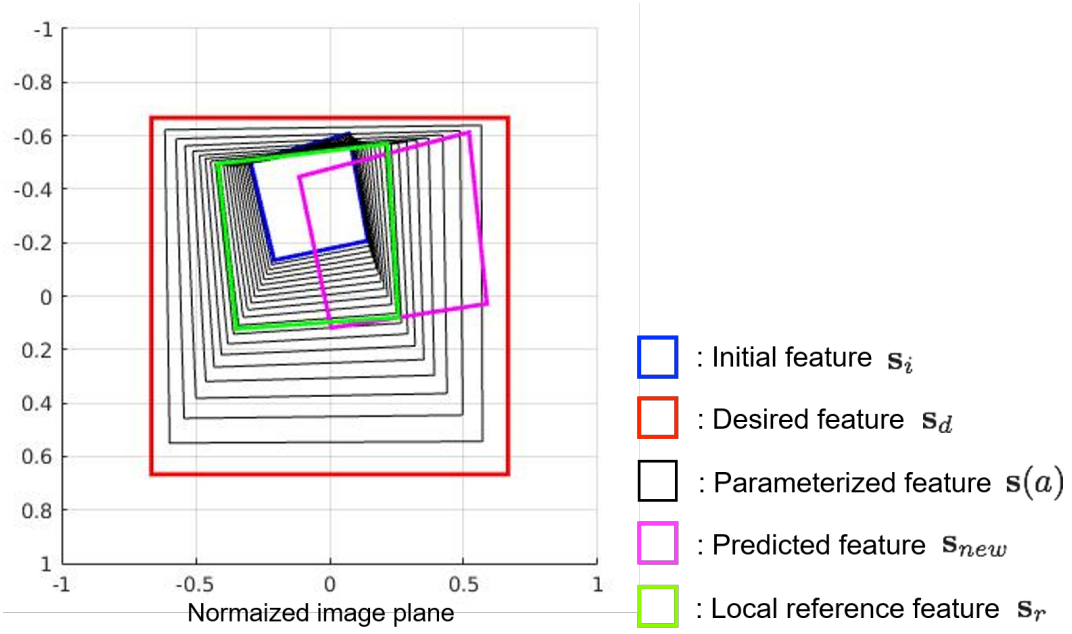


Figure 3.3: Graphical representation of the several features in normalized image plane

4

VPC Configuration

Visual predictive control is consist of prediction model and optimization. In this chapter, each component of VPC is explained in detail. In Section 4.1, the prediction model used in this thesis is presented. Section 4.2 is about the cost function to be optimized. Formulation of the cost function is mainly discussed, and then derivation of its Jacobian is followed. Overall optimization setup is explained in Section 4.3.

4.1 Prediction model

We use the linearized local model (LM) which is proposed in the work of G. Allibert, *et al* [18].

$$\mathbf{s}_{k+1} = \mathbf{s}_k + \dot{\mathbf{s}}_k dt_p \tag{4.1}$$

$$\dot{\mathbf{s}}_k = L(\mathbf{s}_k)\mathbf{u}_k \tag{4.2}$$

Graphical representation of the prediction model is depicted in Fig 4.1. Prediction starts from the current measurement of the feature, so $\mathbf{s}_0$ is constant during the prediction. In the following of this chapter, subscript k refers k-th predicted value. So, $\mathbf{s}_k$ represents a k-th predicted feature, and $\mathbf{u}_k$ represents a k-th predicted guidance command, i.e. body velocity of camera. Also, dt_p refers the prediction time step. Matrix L refers an image Jacobian, or namely interaction matrix,

presented in the work of F. Chaumette, *et al* [1].

$$L(\mathbf{s}) = \begin{bmatrix} -1/Z & 0 & x/Z & xy & -(1+x^2) & y \\ 0 & -1/Z & y/Z & 1+y^2 & -xy & -x \end{bmatrix} \tag{4.3}$$

Note that (4.3) is an interaction matrix of a single point feature. Interaction matrix for four image point is just vertical stack of the each interaction matrix. To calculate interaction matrix exactly, depth of the feature point is required. Moreover, depth of the feature must be differed as prediction proceeded. In this thesis, we fixe depth of the feature to some constant, during the feature prediction step. This assumption is based on movement of the camera in the prediction step is much smaller than the depth of the feature. Therefore, the interaction matrix in (4.2) is unchanged during the prediction, and the prediction model (4.2) is just remained as a function of guidance command, $\mathbf{u}_k$.

In addition to above modification, it is assumed that the predicted guidance command is unchanged during the feature prediction step. By applying (4.4), computational complexity in derivation of the cost function gradient is greatly reduced. This aspect would be re-visited in the following section.

$$\mathbf{u}_k = \mathbf{u}_{k+1} = \bar{\mathbf{u}} \tag{4.4}$$

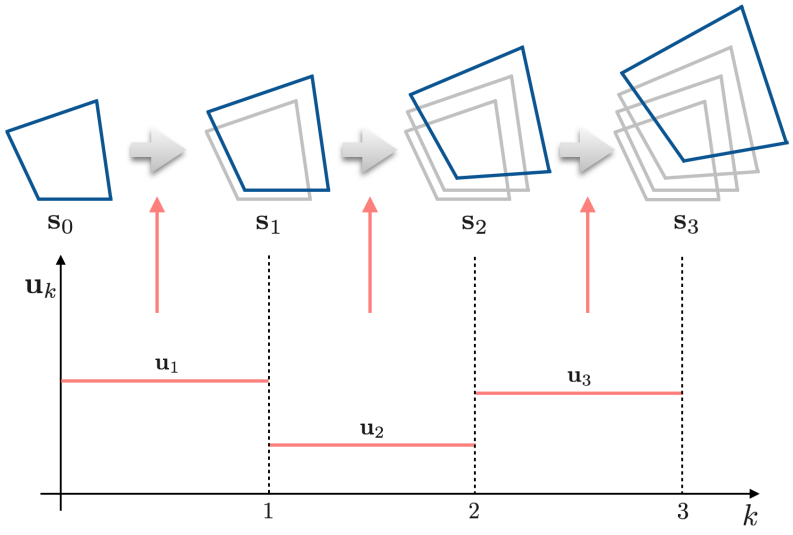


Figure 4.1: Graphical representation of the prediction model in VPC

4.2 Cost formulation

Cost function is key component of the optimization, and we formulate the cost function as like below.

$$J = \frac{1}{2}\sum_{k=1}^{N_p}\left(\rho_d(\mathbf{s}_k - \mathbf{s}_d)^\top(\mathbf{s}_k - \mathbf{s}_d) + \rho_r(\mathbf{s}_k - \mathbf{s}_{r,k})^\top(\mathbf{s}_k - \mathbf{s}_{r,k}) + \rho_u \mathbf{u}_k^\top \mathbf{u}_k\right) \tag{4.5}$$

Here, N_p is a prediction horizon of the prediction model. The above cost function is consist of three terms, and each of the term is weighted by premultiplied factor ρ_d, ρ_r, and ρ_u. The first term in (4.5) is related with penalizing terminal feature error. This term is frequently used in the most of previous VPC works. The second term in the above equation is designed to follow the local reference feature $\mathbf{s}_{r,k}$, which is discussed in the previous chapter. Here, $\mathbf{s}_{r,k}$ refers to the local reference feature corresponding to the k-th predicted feature. This additional term is the key component of the proposed VPC formulation. The last term of the cost function is related with regulation of the optimal guidance command. Role of the regulating term is preventing the guidance command from drastic alteration, so that smooth guidance velocity should be guaranteed.

In the rest of this section, gradient of the cost function is derived. Because main drawback of model predictive control is its slow calculation time, it is highly recommended to utilize analytic gradient of the cost for reducing the number of iteration in the optimization. From (4.1) and (4.2), gradient of the cost function is directly related with Jacobian of the predicted feature with respect to the guidance command.

$$\frac{\partial \mathbf{s}_{k+1}}{\partial \mathbf{u}_k} = \frac{\partial \mathbf{s}_k}{\partial \mathbf{u}_k} + \left(\frac{\partial}{\partial \mathbf{u}_k} L(\mathbf{s}_k)\mathbf{u}_k\right) dt_p \tag{4.6}$$

Equation $(4, 6)$ represents Jacobian of the $(k+1)$-th predicted feature with respect to the k-th predicted command. Similar with the prediction model, Jacobian of the $(k+1)$-th predicted feature is determined from the Jacobian of previous prediction. In other words, deviation of the guidance command at the initial step must affect the Jacobian of the predicted feature at k-th step. This past-dependent prediction procedure makes calculation of the Jacobian much complex. However, by applying (4.4), computation complexity is highly decreased because we simply consider the initial step of the guidance command only. Alternative to say, the Jacobian of the k-th predicted feature with respect to the $k \geq 2$ steps of the guidance command could be ignored. Therefore, we

use (4.7) instead of (4.6), under the condition of (4.4).

$$\frac{\partial \mathbf{s}_{k+1}}{\partial \bar{\mathbf{u}}} = \frac{\partial \mathbf{s}_k}{\partial \bar{\mathbf{u}}} + \left(\frac{\partial}{\partial \bar{\mathbf{u}}} L(\mathbf{s}_k)\bar{\mathbf{u}} \right) dt_p \tag{4.7}$$

Second term of (4.7) is highly coupled with $\bar{\mathbf{u}}$. It could be calculated as following equations.

$$\left[\frac{\partial}{\partial \bar{\mathbf{u}}} L(\mathbf{s}_k)\bar{\mathbf{u}} \right]_j = \left(\frac{\partial}{\partial \bar{u}_j} L(\mathbf{s}_k) \right) \bar{\mathbf{u}} + [L(\mathbf{s}_k)]_j \tag{4.8}$$

$$\frac{\partial}{\partial \bar{u}_j} L(\mathbf{s}_k) = \begin{bmatrix} 0 & 0 & \frac{1}{Z}\frac{\partial x_k}{\partial \bar{u}_j} & \frac{\partial x_k}{\partial \bar{u}_j} y_k + \frac{\partial y_k}{\partial \bar{u}_j} x_k & -2\frac{\partial x_k}{\partial \bar{u}_j} x_k & \frac{\partial y_k}{\partial \bar{u}_j} \\ 0 & 0 & \frac{1}{Z}\frac{\partial y_k}{\partial \bar{u}_j} & 2\frac{\partial y_k}{\partial \bar{u}_j} y_k & -\left(\frac{\partial x_k}{\partial \bar{u}_j} y_k + \frac{\partial y_k}{\partial \bar{u}_j} x_k \right) & -\frac{\partial x_k}{\partial \bar{u}_j} \end{bmatrix} \tag{4.9}$$

Jacobian of the local reference feature $\mathbf{s}_{r,k}$ is complicate to compute. Since the local reference feature is determined from the variable a, and this variable is calculated from the predicted feature $\mathbf{s}_k$. Therefore, chain rule must be used to derive the Jacobian of the local reference feature.

$$\frac{\partial \mathbf{s}_{r,k}}{\partial \bar{\mathbf{u}}} = \frac{\partial \mathbf{s}_{r,k}}{\partial a} \frac{\partial a}{\partial \mathbf{s}_k} \frac{\partial \mathbf{s}_k}{\partial \bar{\mathbf{u}}} \tag{4.10}$$

Note that derivative of the parameterized feature with respect to the variable a, and gradient of the variable a with respect to the predicted feature are already derived in Chapter 3. So, above equation could be calculated by using (3.23), (3.27), and (4.7).

4.3 Optimization setup

In our optimization, two kinds of input constraints are considered. One of them is discussed in the section 4.1, equation (4.4). The other constraint is upper and lower bound of the guidance command. It could be represented as below.

$$\mathbf{u}_{lb} \leq \bar{\mathbf{u}} \leq \mathbf{u}_{ub} \tag{4.11}$$

Where $\mathbf{u}_{lb}$ and $\mathbf{u}_{ub}$ refer to the lower and upper bound of the guidance command, respectively.

We utilized open-source optimization solver to calculate the optimal guidance command. In the simulation environments, MATLAB optimization toolbox is used. Specifically, 'fmincon' function with 'interior-point' algorithm is used. However, this toolbox cannot be appropriate for real-time experiment, because of its long computation time. So, 'nlopt' is used for the experiment, which is also open-source optimization library written in the C language. More specifically, gradient-based 'SLSQP' algorithm is selected for the entire experiment.

Cost/Constraint	**Equation**	**Description**
Terminal error penalizing cost	$\frac{1}{2}\rho_d(\mathbf{s}_k - \mathbf{s}_d)^\top(\mathbf{s}_k - \mathbf{s}_d)$	Distance between the predicted feature and the desired feature
Local reference tracking cost	$\frac{1}{2}\rho_r(\mathbf{s}_k - \mathbf{s}_{r,k})^\top(\mathbf{s}_k - \mathbf{s}_{r,k})$	Distance between the predicted feature and the local reference feature
Input regulating cost	$\frac{1}{2}\rho_u\mathbf{u}_k^\top\mathbf{u}_k$	Norm square of the predicted guidance command
Bounded input constraint	$\mathbf{u}_{lb} \leq \bar{\mathbf{u}} \leq \mathbf{u}_{ub}$	The optimal guidance command must be inside of the admissible region
Consistent prediction constraint	$\mathbf{u}_k = \mathbf{u}_{k+1} = \bar{\mathbf{u}}$	The optimal guidance command is not changed during the prediction

Table 4.1: Organization of the cost terms and constraints in the proposed VPC

5 Simulations

5.1 Simulation settings

In Fig 5.1, simulation environment is depicted in the left figure. Grey objects denote location of the initial and the desired camera, and its normalized image planes are illustrated as transparent rectangle, respectively. The visual target is drawn as black solid line and projected image of the visual target is also drawn in the normalized image plane. Right figure presents image at the initial pose and the desired pose in the same image plane. A blue rectangle represents image of the visual target at the initial pose, and a red rectangle represents image of the visual target at the desired pose.

Two simulations with different setting of parameters are performed. In the simulation case 1, weight of the reference tracking term is set to zero, so only terminal penalizing term in (4.5) takes effect in steering the initial camera to the desired pose. Compared to the first case, the reference tracking weight is set to 2000 in the second case of simulation. By comparing the results of two simulations, effectiveness of tracking the local reference feature could be found. Parameters used in the simulations are listed in Table 5.1.

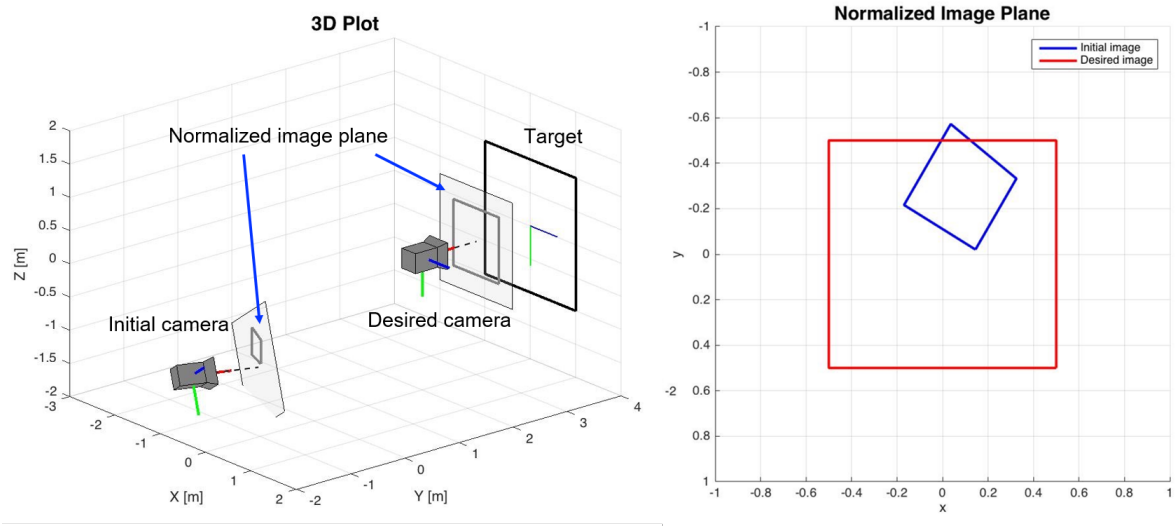


Figure 5.1: Environmental setup for the simulations. (left) location of the camera and the visual target. (right) corresponding normalized image plane.

Parameter	**Value**	
	Case1	Case2
Terminal penalizing weight (ρ_d)	10	10
Local reference tracking weight (ρ_r)	0	2000
Input regulating weight (ρ_u)	0.1	0.1
Prediction horizon (N_p)	3	3
Prediction time step (dt_p)	0.05 s	0.05 s
Upper bound of the command ($\mathbf{u}_{ub}$)	0.5·$\mathbf{1}_{6\times1}$	0.5·$\mathbf{1}_{6\times1}$
Lower bound of the command ($\mathbf{u}_{lb}$)	-0.5·$\mathbf{1}_{6\times1}$	-0.5·$\mathbf{1}_{6\times1}$

Table 5.1: Parameter settings in the simulations

5.2 Simulation results

Overall simulation results are illustrated in Fig 5.2. Results of the first case is located in the left side of the figure (a, c, e), and results of the second case is located in the right side of the figure (b, d, f). Each row of the figure represents the feature trajectory in image plane (a, b), camera trajectory in 3D space (c, d), and optimal guidance command (e, f), respectively.

By comparing the feature trajectory results (Fig 5.2.a, b), it is definitely clear that the reference feature trajectory is properly tracked in the second case of the simulation as shown in Fig 5.2.b. Contrast to the second case, the feature has some tendency of moving to the center of the image plane, at the initial of guidance process in the first case of the simulation. Moreover, the feature is suddenly rotated when the current feature is close to the desired feature as illustrated in Fig 5.2.a. This is because the first case is designed to simply minimize the distance between the current and the desired feature. This simplicity leads to the additional camera motion when the camera is in the vicinity of the desired pose.

In the context of optimal guidance command, clear difference between the both cases could be found. In Fig 5.2.f, the optimal command is uniformly generated from the initial to 6 seconds. But in Fig 5.2.e, some unnatural guidance signal is generated at 6 seconds. This is closely related to the sudden rotation of the feature, as mentioned in the previous paragraph.

This aspect is also found in the camera trajectory graph. Fig 5.2.c shows that the camera tends to drift away from the geodesic path and then converged to the desired position. Compared to the first case, the trajectory of the camera is fairy similar with the geodesic path as illustrated in the Fig 5.2.d. To quantitatively measure the similarity between the camera trajectory and the geodesic path, length of the trajectory is listed in the Table 5.2. The proposed case showed similar trajectory length compared to the geodesic path, as expected.

	Camera trajectory length
Case 1	2.0407 m
Case 2	1.8489 m
Geodesic	1.8371 m

Table 5.2: Comparison of camera trajectory length in the simulations

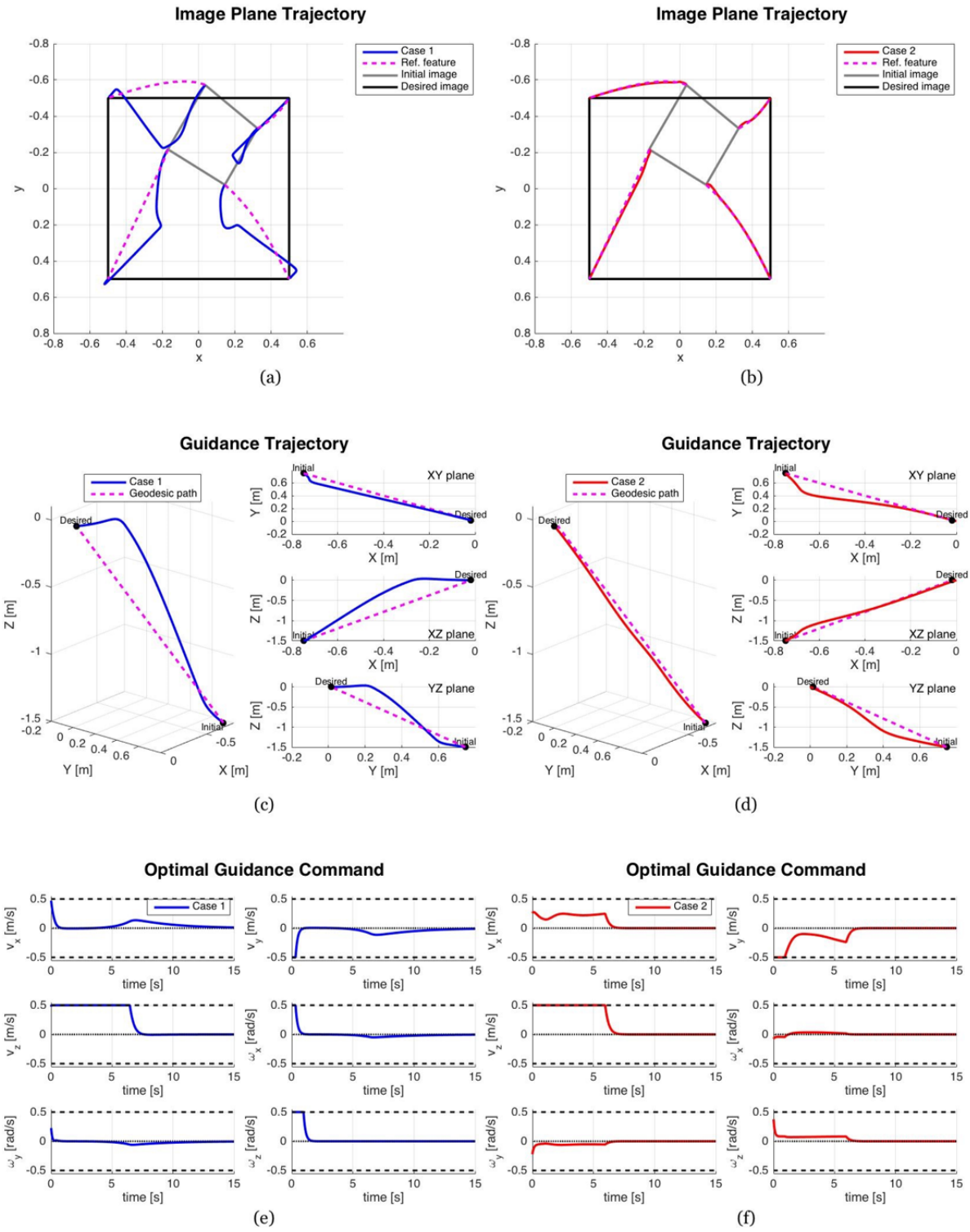


Figure 5.2: Comparative results of the simulations. (a), (b) trajectory of the feature in image plane. (c), (d) trajectory of the camera in 3D space. (e), (f) optimal guidance command history.

6

Experiments

6.1 Experiment settings

Experimental settings are illustrated in Fig 6.1. Four different colors of LED lights are used as the visual target. To detect the visual target, a camera is installed at the end-effector of the manipulator, which has six actuators. Also, this camera captures 640 $\times$ 480 image with 30 fps. Since the optimal guidance command is derived from four image points, suitable image processing steps for extracting feature points are required. The image processing loop runs at 15 Hz on the laptop computer. When the feature extraction is finished, coordinate of the four image points are transfered to an optimization engine. Average optimal input calculation time is about 200 milliseconds. The calculated command is converted to the joint velocity of the manipulator, and then transfered to the manipulator via serial communication. Initial and desired configurations of the manipulator are illustrated in Fig 6.2.

In the experiment, prediction time step in the model prediction process must be determined conservatively, considering the calculation time of the optimization. If the time required for the optimization is longer than the prediction time step, the optimization process could not reflect conditions over the prediction. Therefore, results of the VPC are not applicable to real-time situation, and unexpected motion of the camera could be occur when the total prediction time is shorter than the VPC calculation time.

To determine suitable prediction time step of the VPC, Monte-Carlo method is performed. With respect to some different prediction time step, the optimal guidance calculation time is repeatedly measured. The results are illustrated in Fig 6.3. As the prediction time step becomes longer, the time required for the VPC and its variance are also increased. From this observation, the prediction time step for the VPC is selected to 1.0 seconds. This is because 0.5 seconds of the prediction time step has some possibility of uncovering the optimization calculation time, and 1.5 seconds of the prediction time step is fairly extravagant.

Similar to the simulations, two experiments with different setting of parameters are performed. In the case 1, only the terminal penalizing term is valid for the VPC. However, in the case 2, the reference tracking term as well as the terminal penalizing term takes effect. Therefore, strength of the proposed VPC formulation could be validated by comparing the results of the experiments. Other parameters used in the experiments are listed in Table 6.1. For convenience, only one step of the prediction horizon is considered.

Figure 6.1: Experimental setup

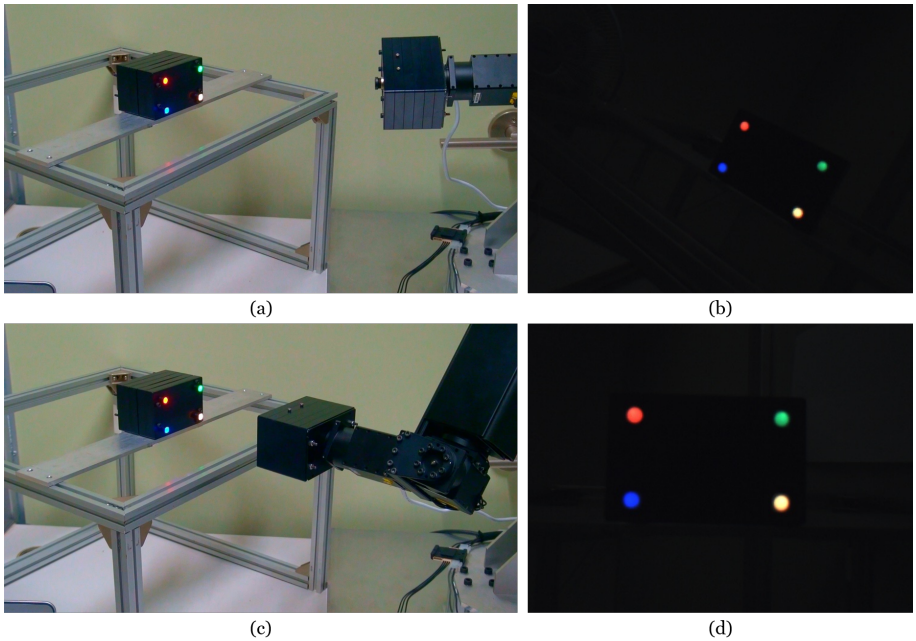


Figure 6.2: Initial and desired configuration of the manipulator. (a), (b) initial configuration and corresponding image captured by the camera. (c), (d) desired configuration and corresponding image captured by the camera.

Parameter	**Value**	
	Case1	Case2
Terminal penalizing weight (ρ_d)	10	0
Local reference tracking weight (ρ_r)	0	400
Input regulating weight (ρ_u)	0.1	0.1
Prediction horizon (N_p)	1	1
Prediction time step (dt_p)	1.0 s	1.0 s
Upper bound of the command ($\mathbf{u}_{ub}$)	0.05·$\mathbf{1}_{6\times1}$	0.05·$\mathbf{1}_{6\times1}$
Lower bound of the command ($\mathbf{u}_{lb}$)	-0.05·$\mathbf{1}_{6\times1}$	-0.05·$\mathbf{1}_{6\times1}$

Table 6.1: Parameter settings in the experiments

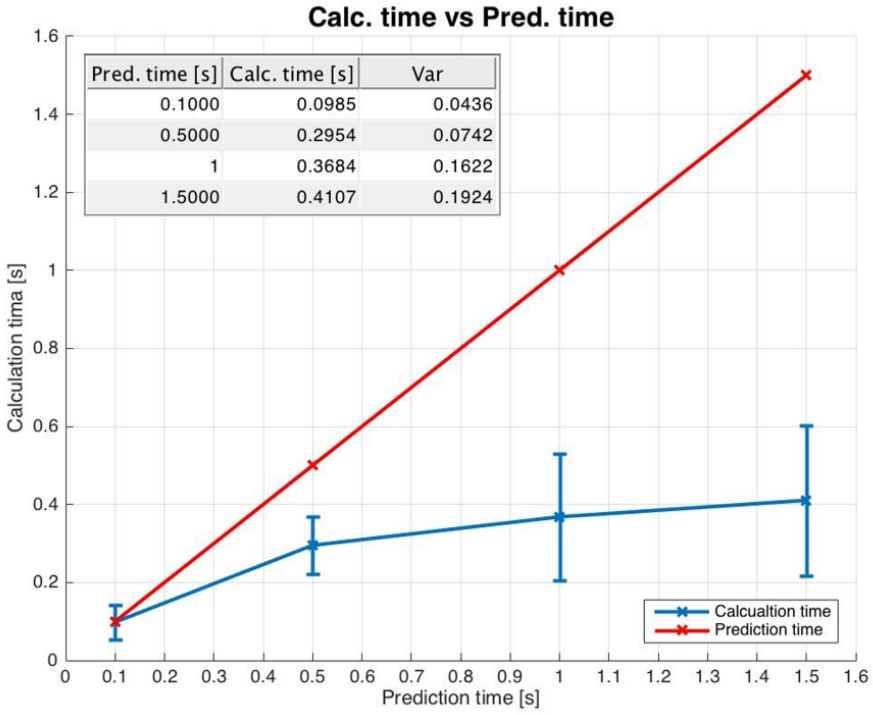


Figure 6.3: Monte-Carlo result of the optimization calculation time. Vertical bar means variance of the calculation time.

6.2 Experiment results

In Fig 6.4, process of the guidance is illustrated for some moments of the experiments. Left column of the Fig 6.4 is procedure of the first case, right column of the figure is procedure of the second case, and rows of the figure are correspond to the progress of the experiments. Captured raw image and history of the feature points are depicted in the left bottom of the each figure. Also, dashed line in the feature history plot represent the reference feature trajectory generated by the method discussed in the chapter 3.

Although two cases are started from the same initial configuration, manipulators showed slight different configuration during the guidance procedure. The most significant difference could be found in Fig 6.4.(e) and (f). Configuration of the two manipulator is clearly distinct, and corresponding location of the image points are also dissimilar. This is because the camera in the first case is just steered in order to move the current features toward the center of the image plane, as stated in the section 5.2. Consequently, the camera in the case 1 moved fairly different direction compared to previous direction so that the current feature could converge to the desired feature at 13 seconds. however, the camera in the case 2 smoothly converged to the desired position.

Other results are illustrated in Fig 6.5, as similar with the Fig 5.2. In optimal guidance command graph of the Fig 6.5, v_z and ω_x showed distinctive difference. In the first case, large value of v_z is exerted compared to the second case at the initial step. This is because the first case has some tendency of moving current feature to the center of the image plane, as stated in the section 5.2. Sequently, large value of ω_x command is required for the camera in the case 1, in order to rotate the current feature to converge to the desired feature.

The resultant guidance trajectories in 3D space directly demonstrate the effectiveness of the proposed algorithm. In Fig 6.5 (c), the trajectory of the camera fairly differ from the geodesic path, which means unnecessary motion of the camera is included. However, the trajectory of the camera in the case 2 is similar to the geodesic path, as illustrated in Fig 6.5 (d). Length of the camera trajectory and the geodesic path is listed in Table 6.2. Although the length of the camera trajectory in the case 2 is not exactly similar with the length of the geodesic path, the proposed algorithm is capable of reducing the travel distance, and steering the camera toward the geodesic path.

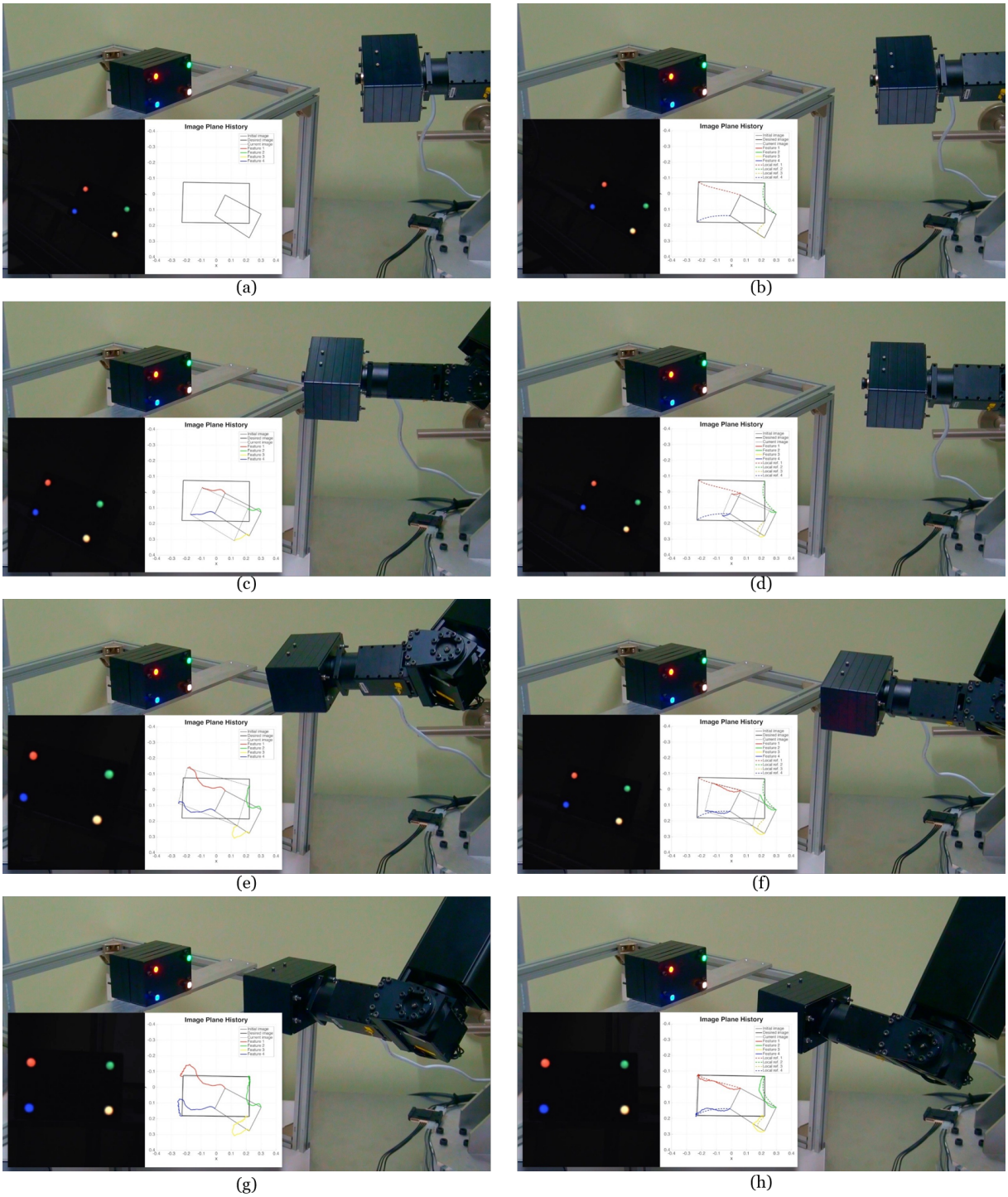


Figure 6.4: Experiment results with respect to time. (a), (c), (e), (g) results of the case 1. (b), (d), (f), (h) results of the case 2. (a), (b) results at $0s$. (c), (d) results at $6s$. (e), (f) results at $13s$. (g), (h) results at $20s$.

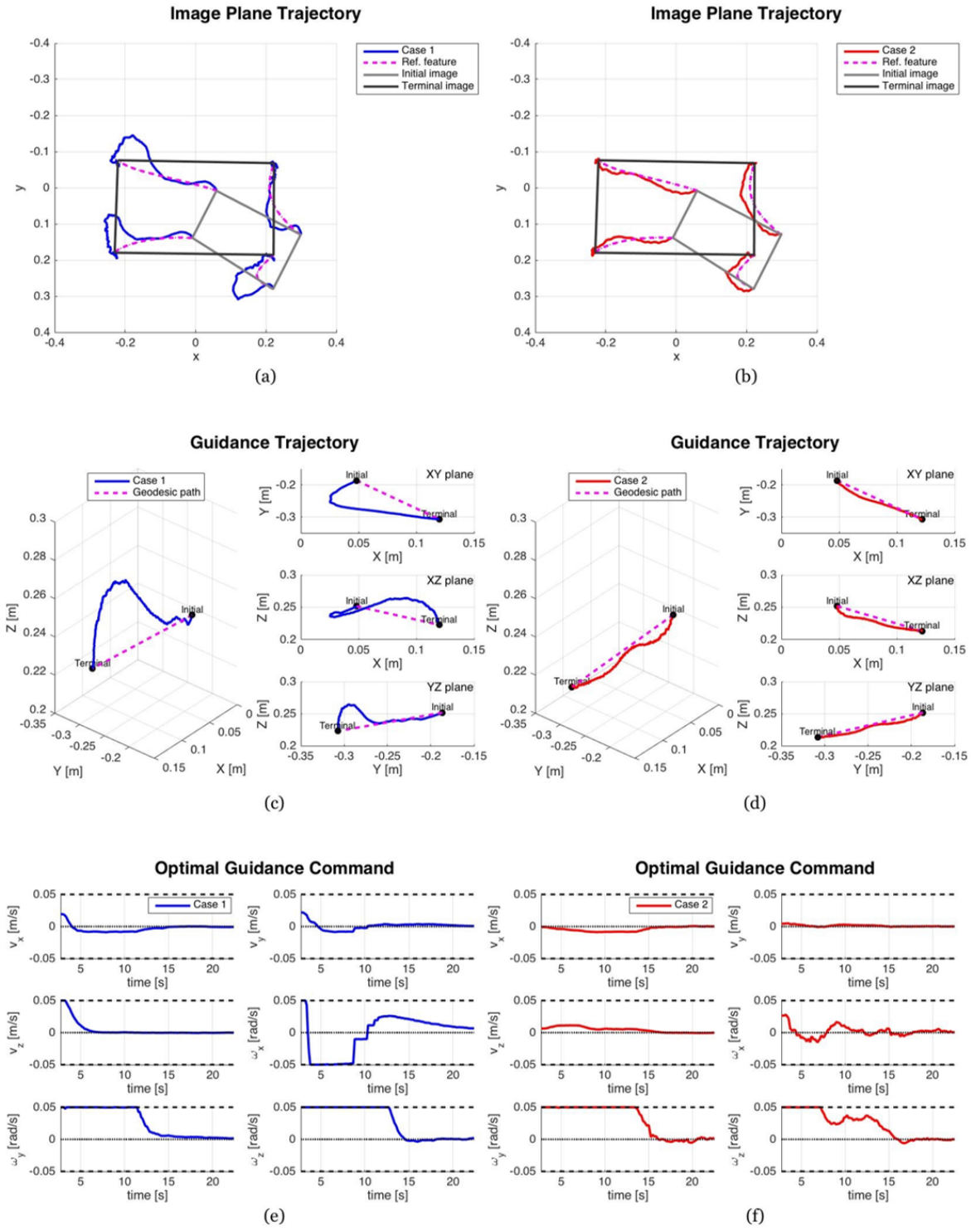


Figure 6.5: Comparative results of the experiments. (a), (b) trajectory of the feature in image plane. (c), (d) trajectory of the camera in 3D space. (e), (f) optimal guidance command history.

Fig 6.6 is about comparison of the optimal input calculation time during the experiments. In the second case, computational burden is slightly harder than the first case. This is because the local reference feature must be calculated whenever feature prediction is performed. However, calculating the local reference feature does not require highly burden process, so difference between the two guidance frequencies are acceptable. Therefore, the proposed VPC could make the guidance trajectory as straight as possible, without much of computational complexity.

	Camera trajectory length
Case 1	0.2683 m
Case 2	0.1899 m
Geodesic	0.1470 m

Table 6.2: Comparison of camera trajectory length in the experiments

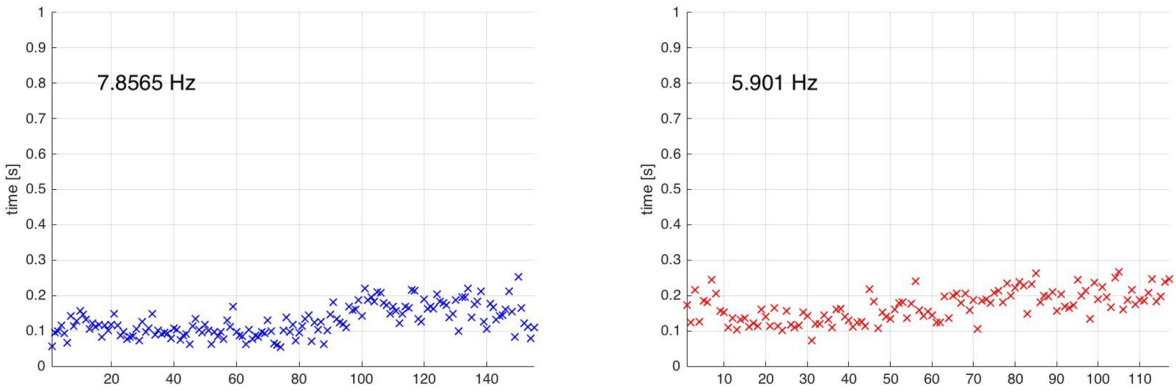


Figure 6.6: Comparision of the VPC calculation time. (left) calculation time of the optimization in the case 1. (right) calculation time of the optimization in the case 2.

7

Conclusion

In this thesis, we performed the model prediction based visual guidance in order to steer the initial camera toward the desired pose as direct as possible. Without metric information of the visual target and depth of the feature, time-dependent reference feature trajectory could be unfeasible. So, we proposed the reference feature trajectory which is parameterized with time-independent scalar variable. The feature parameterization is based on the linear interpolation of partial pose between the initial and the desired camera frame. We also constructed the method to calculate partial pose between the initial and the desired frame even metric information of the visual target is unknown. After the reference feature trajectory is generated, optimization process for the VPC is prepared. The prediction model of the VPC is linearized model (LM), with invariant feature depth assumption during the prediction. We added the reference feature tracking term in the cost function of the VPC so that the camera could follow the generated reference trajectory. To compute the tracking cost, the closest reference feature to the predicted feature is designated to the local reference feature. The proposed optimization problem is solved by several open-source solver with the gradient-based algorithm. The simulation results showed that our VPC could achieve more direct guidance trajectory. The experimental result also demonstrated the effectiveness of the proposed algorithm, with reasonable additional computation time.

Our potential future work is applying our visual predictive guidance to aerial manipulation

task. For the precise manipulation, guidance command must be robust under shaking movement of the aerial platform. Moreover, many constraints such as target visibility, velocity limit, and occlusion from obstacle are must be considered during the manipulation tasks. In this context, VPC is appropriate strategy because it could handle complex constraints through constrained optimization. The main drawback of the VPC is its calculation time, so we planned to re-formulate our cost function of the VPC in order to utilize convex optimization technique, which is greatly faster than general constrained optimization.

References

[1] F. Chaumette and S. Hutchinson, "Visual servo control I: Basic approaches," *IEEE Robot. Automat. Mag.*, vol. 13, no. 4, pp. 82-90, 2006.

[2] F. Chaumette and S. Hutchinson, "Visual servo control II: Advanced approaches," *IEEE Robot. Automat. Mag.*, vol. 14, no. 1, pp. 109-118, 2007.

[3] E. Malis, F. Chaumette, and S. Boudet, "2-1/2 D visual servoing," *IEEE Trans. Robot. Automat.*, vol. 15, no. 2, pp. 238-250, 1999.

[4] F. Chaumette, "Image Moments: A General and Useful Set of Features for Visual Servoing," *IEEE Trans. Robot.*, vol. 20, no. 4, pp. 713-723, 2004.

[5] O. Tahri and F. Chaumette, "Point-Based and Region-Based Image Moments for Visual Servoing of Planar Objects," *IEEE Trans. Robot.*, vol. 21, no. 6, pp. 1116-1127, 2005.

[6] M. Bakthavatchalam and F. Chaumette, "Photometric moments: New promising candidates for visual servoing," *IEEE Intern. Conf. Robot. Automat.*, Shanghai, China, pp. 5241-5246, May, 2011.

[7] A. Dame and E. Marchand, "Entropy-Based Visual Servoing," *IEEE Intern. Conf. Robot. Automat.*, Kobe, Japan, pp. 707-713, May, 2009.

[8] A. Dame and E. Marchand, "Mutual Information-Based Visual Servoing," *IEEE Trans. Robot.*, vol. 27, no. 5, pp. 958-969, 2011.

[9] G. L. Mariottini, G. Oriolo, and D. Prattichizzo, "Image-Based Visual Servoing for Nonholonomic Mobile Robots Using Epipolar Geometry," *IEEE Trans. Robot.*, vol. 23, no. 1, pp. 87-100, 2007.

[10] G. Lopez-Nicolas, J. J. Guerrero, C. Sagues, "Visual control through the trifocal tensor for nonholonomic robots," *Robotics and Autonomous Systems*, vol. 58, no. 2, pp. 261-226, 2010.

[11] A. Shademan and M. Jagersand, "Three-View Uncalibrated Visual Servoing," *IEEE Intern. Conf. Intell. Robot. Syst.*, Taipei, Taiwan, pp. 6234-6239, Oct., 2010.

[12] G. Chesi and Y. S. Hung, "Global Path-Planning for Constrained and Optimal Visual Servoing," *IEEE Trans. Robot.*, vol. 23, no. 5, pp. 1050-1060, 2007

[13] G. Chesi, "Visual servoing path planning via homogeneous forms and LMI optimizations," *IEEE Trans. Robot.*, vol. 25, no. 2, pp. 281-291, 2009.

[14] Y. Mezouar, F. Chaumette, "Path planning for robust image-based control," *IEEE Trans. Robot. Automat.*, vol. 18, no. 4, pp. 534-549, 2002.

[15] M. Kazemi, K. Gupta, and M. Mehrandezh, "Global path planning for robust visual servoing in complex environments," *IEEE Intern. Conf. Robot. Automat.*, Kobe, Japan, pp. 326-332, May, 2009.

[16] A. Shademan and M. Jagersand, "Robust Sampling-Based Planning for Uncalibrated Visual Servoing," *IEEE Intern. Conf. Intell. Robot. Syst.*, Vila Moura, Portugal, pp. 2663-2669, Oct., 2012.

[17] M. Sauvee, P. Poignet, E. Dombre, E. Courtial, "Image based visual servoing through non-linear model predictive control," *IEEE Conf. Deci. Cont.*, CA, USA, pp. 1776-1781, Dec., 2006.

[18] G. Allibert, E. Courtial, and F. Chaumette, "Predictive control for constrained image-based visual servoing," *IEEE Trans. Robot.*, vol. 26, no. 5, pp. 933-939, 2010.

[19] A. Assa, F. Janabi-Sharifi, "Robust model predictive control for visual servoing," *IEEE Intern. Conf. Intell. Robot. Syst.*, IL, USA, pp. 2715-2720, Oct., 2014.

[20] A. Assa and F. Janabi-Sharifi, "Hybrid Predictive Control for Constrained Visual Servoing," *IEEE Intern. Conf. Advan. Intell. Mech.*, Besancon, France, pp. 931-936, July, 2014.

[21] A. Burlacu, and C. Lazar, "Reference trajectory-based visual predictive control," *Advanced Robotics*, vol. 26, no. 8, pp. 1035-1054, 2012.

[22] A. Burlacu, C. Copot, E. Cervera, and C. Lazar, "Real-time visual predictive control of manipulation systems," *IEEE Intern. Conf. Robot. Automat.*, Tallinn, Estonia, pp. 383-388, May, 2011.

[23] D. Lee, H. Lim, and H. Jin Kim, "Obstacle Avoidance Using Image-based Visual Servoing Integrated with Nonlinear Model Predictive Control," *IEEE Conf. Deci. Cont.*, FL, USA, pp. 5689-5694, Dec., 2011.

국 문 초 록

영상기반 유도의 목적은 카메라를 목표 지점으로 이동시키기 위한 유도 명령을 설계하는 것이다. 본 논문에서는 영상기반 모델 예측 기법을 사용하여 직선 궤적을 그리며 목표 자세로 이동하기 위한 카메라의 유도 명령을 생성하는 것에 초점을 두고 있다. 직선의 유도 궤적을 목표로 함으로써 카메라의 불필요한 움직임을 최소화 할 수 있다. 그리고 모델 예측 기법은 시스템 제약조건을 고려할 수 있고, 스케일의 부정확성에 강건하다는 장점이 있기 때문에 영상기반의 유도 명령을 설계하는데에 적합하다. 따라서 본 논문에서는 목표 이미지와 초기 이미지로부터 부분 자세추정을 수행하고 이를 선형보간하여 기준 피쳐 궤적을 설계하였다. 또한 매개변수화된 기준 피쳐 궤적을 모델 예측 기법의 예측 모델에서 고려하기 위해 로컬 기준 피쳐를 정의하였다. 나아가 본 논문에서는 생성된 기준 피쳐 궤적을 추종하기 위한 성능지수를 로컬 기준 피쳐로부터 정의하고 이를 기반으로 모델 예측 제어를 수행하였다. 예측 모델에서 특징점의 깊이가 변하지 않는다는 가정을 추가함으로써 최적화 과정의 계산시간을 줄일 수 있었다. 본 연구에서 제안한 영상기반의 유도 기법은 추가적인 계산을 크게 요구하지 않으면서 직선에 가까운 유도궤적을 형성할 수 있음을 시뮬레이션과 실험을 통해 검증하였다.

주요어 : 영상기반 유도, 영상 예측 제어, 예측 모델, 최적화

학번 : 2014-20672