CAMERA MOTION BLUR AND ITS EFFECT ON FEATURE DETECTORS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

FERİT ÜZER

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2010

Approval of the thesis:

CAMERA MOTION BLUR AND ITS EFFECT ON FEATURE DETECTORS

submitted by **FERİT ÜZER** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. İsmet Erkmen
Head of Department, **Electrical and Electronics Engineering** _____

Asst. Prof. Dr. Afşar Saranlı
Supervisor, **Electrical and Electronics Engineering Dept., METU** _____

Examining Committee Members:

Prof. Dr. Kemal Leblebicioğlu
Electrical and Electronics Engineering Dept., METU _____

Asst. Prof. Dr. Afşar Saranlı
Electrical and Electronics Engineering Dept., METU _____

Prof. Dr. Gözde Bozdağı Akar
Electrical and Electronics Engineering Dept., METU _____

Asst. Prof. Dr. Emre Tuna
Electrical and Electronics Engineering Dept., METU _____

Asst. Prof. Dr. Yiğit Yazıcıoğlu
Mechanical Engineering Dept., METU _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: FERİT ÜZER

Signature :

ABSTRACT

CAMERA MOTION BLUR AND ITS EFFECT ON FEATURE DETECTORS

Üzer, Ferit
M.S., Department of Electrical and Electronics Engineering
Supervisor : Asst. Prof. Dr. Afşar Saranlı

September 2010, 90 pages

Perception, hence the usage of visual sensors is indispensable in mobile and autonomous robotics. Visual sensors such as cameras, rigidly mounted on a robot frame are the most common usage scenario. In this case, the motion of the camera due to the motion of the moving platform as well as the resulting shocks or vibrations causes a number of distortions on video frame sequences. Two most important ones are the frame-to-frame changes of the line-of-sight (LOS) and the presence of motion blur in individual frames. The latter of these two, namely motion blur plays a particularly dominant role in determining the performance of many vision algorithms used in mobile robotics. It is caused by the relative motion between the vision sensor and the scene during the exposure time of the frame. Motion blur is clearly an undesirable phenomenon in computer vision not only because it degrades the quality of images but also causes other feature extraction procedures to degrade or fail. Although there are many studies on feature based tracking, navigation, object recognition algorithms in the computer vision and robotics literature, there is no comprehensive work on the effects of motion blur on different image features and their extraction.

In this thesis, a survey of existing models of motion blur and approaches to motion deblurring

is presented. We review recent literature on motion blur and deblurring and we focus our attention on motion blur induced degradation of a number of popular feature detectors. We investigate and characterize this degradation using video sequences captured by the vision system of a mobile legged robot platform. Harris Corner detector, Canny Edge detector and Scale Invariant Feature Transform (SIFT) are chosen as the popular feature detectors that are most commonly used for mobile robotics applications. The performance degradation of these feature detectors due to motion blur are categorized to analyze the effect of legged locomotion on feature performance for perception. These analysis results are obtained as a first step towards the stabilization and restoration of video sequences captured by our experimental legged robotic platform and towards the development of motion blur robust vision system.

Keywords: motion blur, motion blur models / identification, feature detectors, Harris Corner, Canny Edge, sift, matching.

ÖZ

KAMERA HAREKET BULANIKLIĞI VE ÖZNİTELİK VEKTÖRLERİNE ETKILERI

Üzer, Ferit
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi : Yard. Doç. Dr. Afşar Saranlı

Eylül 2010, 90 sayfa

Algılama bu sebeple görüntüleme sensörlerinin kullanımı mobil ve otonom robotlar da kaçınılmazdır. Robot üzerine monte edilmiş kamera gibi görüntüleme sensörleri çok kullanılan uygulamalardır. Bu durumda hareketli platformun harekinden ve aynı zamanda oluşan şok ve titreşimlerden kaynaklanan kamera hareketi video kareleri üzerinde çeşitli bozulmalara yol açmaktadır. Bunlardan en önemli ikisi videoda bir kareden diğer kareye geçerken ki görüş alanında ki değişiklikler ve video karelerinin her birinde oluşan hareket bulanıklığıdır. Bu ikiliden hareket bulanıklığı olarak adlandırılan etki mobil robotlarda kullanılan görüntü algoritmalarının performans değerlendirmesinde daha baskın ve özel bir rol oynar. Bu etki görüntü sensörü ve sahne arasındaki görüntüleme süresince gerçekleşen görece hareket sebebiyle oluşmaktadır. Hareket bulanıklığı hem görüntü kalitesini düşürmesi hem de görüntü özelliklerini belirleyen yordamların başarısız olmasına sebep olması dolayısıyla bilgisayarla görme alanında istenmeyen bir fenomendir. Robotik ve bilgisayarlı görme literatüründe özellik temelli takip, yer yön belirleme, nesne tanımlama algoritmaları ile ilgili pek çok çalışma olmasına rağmen, hareket bulanıklığının görüntü özellikleri ve onların belirlenmesi üzerindeki etkisi üzerine kapsamlı bir çalışma bulunmamaktadır.

Bu tezde, var olan hareket bulanıklığı modelleri ve bu sorunu giderme yaklaşımları üzerine detaylı bir inceleme yapılmıştır. Hareket bulanıklığı ve bulanıklığı giderme üzerine son yıllarda yapılan çalışmaları inceleyip video karelerinde öznitelik vektörleri çıkarımı üzerinde bulanıklık sebepli oluşan negatif etkiler üzerine odaklandık. Bu negatif etki mobil bacaklı robot platformu üzerinde bulunan görüntüleme sistemi tarafından kaydedilen görüntü serileri kullanılarak incelenip karakterize edilmiştir. Harris köşe algılayıcısı, Canny kenar algılayıcısı ve yerel değişmez öznitelikleri bulan SIFT mobil robotik uygulamalarında sık kullanılan popüler öznitelik algılayıcıları olarak seçilmiştir. Bu öznitelik algılayıcılarının performanslarındaki negatif etki bacaklı hareketin öznitelik algılayıcılarının algılama açısından performansları üzerindeki etkisini analiz etmek için kategorize edilmiştir. Bu analiz sonuçları deneysel bacaklı robot platformu tarafından kaydedilen video serilerinin sabit hale getirilmesi ve yenilenmesi yönündeki ve de hareket bulanıklığına karşı gürbüz bir görüntüleme sistemi geliştirilmesi yönündeki ilk adım olarak görülmektedir.

Anahtar Kelimeler: hareket bulanığı, hareket bulanığı modelleri / tanımlama öznitelik vektörleri, Harris Köşe öznitelik vektörü algılayıcısı, Canny Kenar öznitelik vektörü algılayıcısı, sift, eşleştırme.

to my loving mother, to my beloved fiancée

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my supervisor, Afşar Saranlı, for his guidance, patience, advice, encouragement, and tremendous support throughout my M.S. study. He was always there to listen and give advice when I needed and Without our frequent technical discussions and his supervision during my research, I would never have been able to complete the work presented in this thesis.

I am thankful to all the members of the SensoRHex project. I would like to express my deep and sincere respect to Afşar Saranlı (the project manager), Uluç Saranlı, Yiğit Yazıcıoğlu and Kemal Leblebicioğlu for giving me the opportunity to be a part of this brilliant project environment.

I would also like to express gratitude to (TÜBİTAK) Science Fellowships and Grant Programs Department (BİDEB) for their financial support.

I would also like to give big thanks to all members of Rolab (Laboratory of Robotics and Autonomous Systems) and to my friends Hakan Ertem, Emre Kale, Erinç Altıntaş, Muzaffer Gökhan Güvensen, Tuğcan Aktaş and Selçuk Kavut for their encouragement, suggestions and for their having such friendly environment at METU.

Last but not the least; I owe my loving thanks to my loving mother and my beloved fiancée for their undying love, support and encouragement.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

Motion blur is the result of the relative motion between the camera and the scene over the period of exposure determined by the mechanical or electronic shutter speed. In an image, projection of any point in the scene moving with respect to the camera will be a certain curve on the sensor rather than a point. Thus it will look ”blurred” along the direction of relative motion. In other words, this blurring may appear on the image of a moving object or on the image of a static scene where the camera has moved during capture. The amount of motion blur increases as either the exposure time or the speed of the relative motion between the camera and the scene increases, and it becomes more apparent at higher resolutions since more pixels are effected.

Motion Blur can be used for aesthetic purposes in photography such as for emphasizing the dynamic nature of a scene. For example, using motion blur effect is the common way of showing a sense of speed in sports photography especially in motor sports. It has also been used in computer graphics to create more realistic images because appearance of motion blur is a strong perceptual cue for the illusion of object motion [46], [31], [63], [21], [10], [37], [57]. Moreover, it looks natural in a film or a television image sequence because the human eye behaves in much the same way. Several representations and models for motion blur in human and machine vision have been proposed [1], [23], [27], [42]. Motion blur can also be used to extract motion information or other structure information from individual images and hence can be used for perception. For example, motion blur has been used in the literature to obtain motion and scene 3D structure information [59], [58], [54], [20], [18], [16],

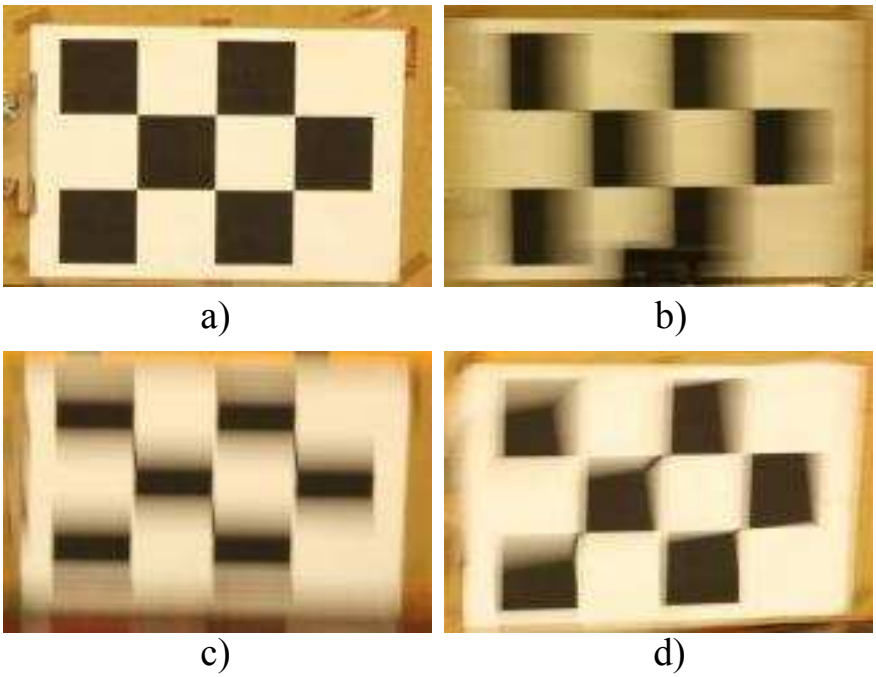


Figure 1.1: The effect of motion blur. Sharp image taken by using tripod is shown at a), horizontal motion blurred image is shown at b), vertical motion blurred image is shown in c) and rotational motion blurred image is shown at d).

[14]. Despite its usefulness to human viewers, motion blur is an extensive image distortion and undesired effect in a large number of other applications. In photography, a perfectly fixed camera and a static scene is necessary to capture the sharpest, most detailed image so motion blur is considered to be an effect that can significantly degrade image quality. Fig. 1.1 demonstrates real examples of images that are blurred by simple linear horizontal and vertical motions. In practice, it is much more complicated and every single image is likely to be uniquely blurred because of the large space of possible motion paths.3

Motion blur degrades and distorts the video frames as well. A video is always expected to be of high-quality so that people feel comfortable when they use it and computer vision algorithms work well on it but the motion blur will spoil this by affecting severely the perception of the frame sequence content. A motion deblurring step is necessary to be applied to video in many applications such as surveillance [28], [12], and the visual tracking applications [29], [30], [27]. In a surveillance system, motion blur deteriorates the effectiveness of object extraction and identification and complicates the task of event detection. Also in the case of visual tracking, rapid camera motions cause image features to move large distances in a single frame. Indeed such distortions may cause the failure of many feature extraction algorithms

such as edge detection [11], corner detection [25] and scale-invariant feature transform (sift) [38], [39], which are used in visual tracking applications. Although these feature detectors are widely used in many robot vision algorithm and well known in literature, there is no specific work which analyzes their performance under motion blur effect.

Perception therefore usage of sensors is indispensable in mobile and autonomous robotics. Vision sensor rigidly mounted on a robot is one of the most common scenario among of the sensors used in robotics. In this case, motion of camera due to the moving platform, shocks or vibrations causes several distortions on video frame sequences. In fact, the oscillation of the line-of-sight (LOS) and the motion blur in frames are these distortions which is common when the visual sensor is mounted on a moving platform such as a mobile robot.

If the motion of the robot is smooth,then the corresponding video sequence recorded by the camera will be smooth too. However, it is almost impossible to obtain smooth motion especially in outdoor and legged robotics due to irregularities of the terrain, obstacles in the way of the robot, friction in the moving parts, vibration induced by motors and legs. All these undesired conditions in the robot motion propagate to the camera, making the camera output very hard for humans to view and to operate or for a robot vision algorithm to process. For this problem, image stabilization defined as the process of removing scene oscillation and making the video sequence less shaky is necessary.

The second distorting effect called as motion blur in each frame is the result of relative motion between the camera and the scene during the exposure time. This relative motion smears the frames and so makes the perception of details more hard as it is discussed in the paragraphs above. The velocity of the robot (imaging platform) and the exposure time of the camera affect the intense and severity of motion blur. Thus, even the smooth component of the motion results in motion blur in frames which is different than the reasons of LOS effect although large vibration amplitude and high frequency increase both effects. For this distortion effect, motion deblurring can be used as a front-end system in a variety of robot vision algorithms or simply as a visualization tool.

The relative effects of these two types of degradations on the ability of human observers to recognize targets are investigated in the work of Adrian Stern [2]. The result of this study clarifies that motion blur has the main reason for perception degradation in the case of severe vibrations. Another work on humanoid robot [47] shows that the classical feature detectors

and descriptors do not work well in presence of motion blur although they are proved to work well for wheeled robot. Therefore, motion blur should be considered and deblurring of the frames should be emphasized when designing dynamic imaging system especially in our case on RHex.

In our research, the effect of motion blur on well-known feature detectors such as Harris corner and Canny edge detectors are investigated on the video sequences as a first step through the stabilization and restoration of video sequences captured by RHex platform. The performance of these detectors are evaluated by running our imaging platform RHex in different velocities. Such an analysis is critical for good experimental practice with the aim of obtaining knowledge about how legged locomotion of our robot propagates to camera and about the degradations on perception of important features in frames. Since feature based algorithms is used in many robot vision algorithms such as visual tracking, navigation etc.

RHex is a robotic platform that consists of a rigid body with six compliant c shaped legs, each possessing one independent actuator. It is designed to have exhibited general mobility over general terrain approaching the complexity and diversity of the natural landscape. In such a platform, it is impossible to minimize shocks and vibrations by proper design due to its priorities on mechanically simple and autonomous design criteria. These shocks and vibrations due to leg locomotion makes the camera output very jittery and blurred that is very hard for human operator and causes robot vision tasks to fail because of the adverse effect of motion blur.

1.2 Methodology and Outline of the Thesis

Our method is based on experimental characterization of motion blur on the most popular and well known feature detectors such as Harris Corner detector, Canny edge detector and SIFT. We investigate and characterize the degradation due to motion blur using video sequences captured by the vision system of a mobile legged robot platform. Although many robot vision algorithms in literature are based on these feature detectors, there is no specific work which analyzes their performance under motion blur effect.

The performance degradation of these well known feature detectors due to motion blur are systematically categorized to analyze the effect of legged locomotion on feature performance

for perception. These analysis results are obtained as a first step towards the stabilization and restoration of video sequences captured by our experimental legged robotic platform.

This thesis consist of 5 chapters. We start with the introduction to the motion blur phenomenon and necessary background in Chapter 2. Later in Section 2.1 and Section 2.2, blur models and blur identification methods are classified. Then, the most popular feature detectors used in robot vision applications such as Harris Corner, Canny Edge Detector and Scale Invariant Feature Detector are given in Chapter 3. The reason why these feature detectors are selected is explained in Section 3.1. The performance degradation of these feature detectors due to motion blur are categorized to analyze the effect of legged locomotion on feature performance for perception. These categories are discussed in Section 3.5

In Chapter 4 a comprehensive evaluation of feature detectors under motion blur is given in Chapter 2. Firstly, experimental scenario is explained and then evaluation criteria is given. These analysis results are presented for each feature detector for each video sequence. The conclusions and future work are given in Chapter 5. All of the algorithms throughout the thesis are implemented on MATLAB.

CHAPTER 2

LITERATURE REVIEW ON MOTION BLUR, BLUR MODELS AND BLUR IDENTIFICATION

As we briefly discussed motion blur is an important problem in computer vision. Therefore, motion deblurring is an inevitable step to increase the success of the computer vision algorithms especially in mobile robotics where the camera is generally exposed to extensive motion.

Motion deblurring can be defined as the deconvolution with a global Point Spread Function (PSF) for images distorted by linear motion or as the deconvolution with a spatially varying PSF for images distorted by more complex motion paths. Moreover, it can be divided in two parts: motion estimation and deconvolution. The first part deals with the challenge to identify the path the camera has followed during the image capture process. The second part uses this information to reverse the convolution during the image formation process in order to restore the sharp picture.

Although our focus is on robot vision, the literature is divided in photography and video capture;

Blind deconvolution can be called as traditional deblurring methods in the literature. These methods obtain an estimate PSF and try to deconvolve the blurred image with that estimated PSF at the same time. Richardson-Lucy [50], [40] and Wiener deconvolution [62] are the notable examples of traditional deblurring. Deblurring images and spectra [24] and the well-known image processing book named as digital image processing [22] provide comprehensive literature survey on these traditional methods.

The main deficiency of these traditional methods is ringing effect in the deblurred images.

Several studies focus on reducing ringing artifacts. Such as total variation regularization is used with Richardson-Lucy algorithm [45] and another example is that gradient sparsity constraints are added to Richardson-Lucy algorithm [34] for the same aim. A multi scale non-blind deconvolution method is another example work [67] to improve the resulting deblurred images.

The main difficulty of this problem comes from PSF estimation or deconvolution step. Therefore, there are also some interesting works which try to make these steps easier. Using supplementary data besides vision data and constraining the deblurring procedure lately become popular in the literature. Using natural image statistic [19] as a constraint in deconvolution problem can be counted as one of the leading example to these methods. Using alpha mask of the blurred region is another interesting idea to make PSF estimation step easier [26]. Another leading and the most interesting idea for our work proposed by Raskar et al. [48] first. Changing the shutter time of a consumer camera according to a predetermined is the main idea of their work. By doing this, it is aimed to obtain better PSF for deblurring process.

There are other methods which use more than one image to get better results in the end of deconvolution step. Bascle et al. Using blurred video input and obtaining a single unblurred high-resolution image [5] is the first example of this idea in the literature. There are also many recent works based on this idea of making the PSF estimation easier. Leading examples are processing one noisy and one blurred image [66] and working on images blurred in horizontal and vertical directions [49]. Extension of [49] is to deblur without orthogonality condition [13]. A recent version of the work of Bascle [5] offers to deblur a video sequence by using high-resolution photographs [8]. However this fails if the scenes are not static. Using a fast low-resolution camera with a high-resolution and slow camera is proposed by Ben-Ezra and Nayar [6], [7]. The main idea of using the extra camera is to obtain accurate PSF estimation and to use it for deblurring high-resolution video sequence. One extension of this work for an other aim is to put one low-resolution and one high-resolution video cameras parallel to each other and run them in different frame rates [36]. The most related work with ours is the work by Agrawal et al. [3] proposed ”PSF null-filling”. In this work, smooth PSFs are obtained by changing the exposure time according to predetermined sequence while video recording procedure is going on.

A global PSF assumption which is not valid for spatially-varying motion blur is the common

feature of the methods mentioned above. Recently, there are also works which deal with spatially-varying case. Such as single image based method make use of image statistics with a stable background assumption [33]. Another idea [4] is to divide an image into small regions which can be assumed to have same PSF or another version of this idea [15] is to work on a pair of blurry images and is to do the local PSFs estimation step and deconvolution step at the same time. For moving object case, the usage of supplementary camera is shown in the work of Ben-Ezra and Nayar [7]. Although these methods deal with spatially-varying case [33], [4], [15], [7], they still work on spatially-invariant local PSF assumption. Without this assumption space-variant blur is considered in the work [53] which focus on only rotational movement of camera. In contrast to this work, space-variant blur is considered for a camera moving without rotation in the work by [55]. Unfortunately, the motion trajectory used in this work is far from the real trajectory of a handheld camera.

In addition to all these remarkable studies, there are interesting works which tries to get rid of motion blur by the fusion of visual and inertial measurements especially in augmented reality literature [64], [65] although their main aim is not to get motion deblurred images or frames. The most recent and interesting work for our case is proposed by Klein and Drummond [29]. They used rotational measurements from rate gyroscopes not only to provide the visual sensor with a pose prediction, but also to modify the operation of the sensor's edge detection.

2.1 Motion Blur Models

First, we need a mathematical model that relates the given blurred image to the unknown "ideal" image in order to deblur an image. An "ideal" image can be considered as one that captures a moment in time instantaneously and therefore with no motion blur. However, it is not possible in practice. The exposure time defines a temporal box filter which causes blur if there is relative motion between camera and scene due to the fact that this box filter destroys important high frequency spatial details.

It is well known that homogeneous blurring which means that blurring is in exactly the same way at every spatial location of image can be defined by convolution in the spatial domain 2.1, 2.2 or by a product operation in the frequency domain 2.3;

$$z = u * h[x, y] = \int u(x - s, y - t)\, h(s, t)\, ds dt \tag{2.1}$$

for discrete case;

$$z = \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} u\,(i-k,\, j-l)\, h\,(k,l) \tag{2.2}$$

$$Z = U\,(w_1, w_2)\,.H\,(w_1, w_2) \tag{2.3}$$

where u is an ideal image, h is called the convolution kernel or point-spread function (PSF) and z is the blurred image. The PSF is an energy density function that describes the amount of time light from a single point in the scene exposes each (x,y) pixel position in the image detector.

This spatially invariant PSF h which models homogeneous blurring should satisfy these;

- The physics of the underlying image formation process h
- If the image is real-valued, then h is also real-valued.
- The PSF h must satisfy the following energy conservation constraint:

$$z = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h\,(x,y)\, dxdy = 1, \tag{2.4}$$

and in discrete case:

$$z = \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} h\,(i,\, j)\, dxdy = 1, \tag{2.5}$$

The simple model can only be used under the homogeneity condition. If the focal length of camera is short or camera rotates significantly about the optical axis, then the intensity of blur changes in the image. In other words, it becomes a complex function of depth of scene and relative motion between camera and scene [55]. Therefore, spatially invariant PSF is not sufficient to model the complex blur caused by leg locomotion. More general linear operation is necessary to define this spatially varying blur.

$$z = u\hat{*}h\,[x,y] = \int u\,(x-s, y-t)\, h\,(x-s, y-t; s, t)\, dsdt \tag{2.6}$$

where u is an ideal image, h is PSF and z is the blurred image again. Note that equation(2.1) is the special case of equation(2.6) in the sense of h does not change with image coordinates x and y. Therefore equation(2.6) can be called as space-variant convolution. If also noise is

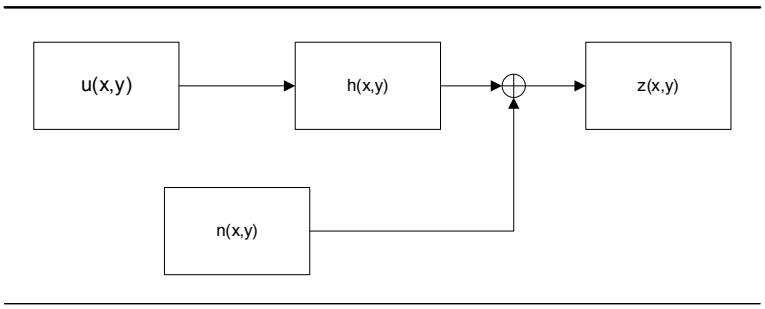


Figure 2.1: The model of motion blur diagram

considered, then complete models of motion blurring is illustrated on figure 2.1. It can be formulated as in 2.7, 2.8.

$$z = u * h\left[x, y\right] + n\left(x, y\right). \tag{2.7}$$

$$z = u\hat{*}h\left[x, y\right] + n\left(x, y\right). \tag{2.8}$$

n(x,y) modeled as an additive term is the noise. Generally, white noise with zero mean is used. It is statistically formulated as in [56];

$$E\left[n\left(x, y\right)\right] \approx \sum_{l_1}^{N-1} \sum_{l_2}^{M-1} n\left(l_1, l_2\right) = 0 \tag{2.9a}$$

$$R_w\left(l_1, l_2\right) \approx \begin{cases} \sigma_n^2 & \text{if } l_1 = l_2 = 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.9b}$$

Motion blur is basically the result of the relative motion between the imaging sensor and the scene over the period of exposure determined by the shutter speed. Therefore, it changes due to the type of motion. This can be in the form of a translation, a rotation, a sudden change of scale, or some combinations of all these and it is quite hard to construct a universal model that covers all of these blur processes. In fact, it is already categorized according to the type of relative motion between camera and scene. In the following the most common motion blur models in the literature will be given.

2.1.1 1D Linear Motion Blur Model

When there is a steady motion between a planar scene perpendicular to the optical axis and the camera in a plane parallel to the scene over the period of exposure $[0, t_{exposure}]$, the resulting psf is a space invariant 1-D rectangular impulse in the direction of motion;

$$h(x,y) = \begin{cases} \frac{1}{L} & \text{if } -\frac{L}{2} \leq x \leq \frac{L}{2} \text{ and } y = 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.10}$$

where the intensity of the PSF $L = v_{relative} t_{exposure}$. When this relative motion makes an angle with the horizontal axis of the scene over the period of exposure;

$$h(x,y) = \begin{cases} \frac{1}{L} & \text{if } \sqrt{x^2 + y^2} \leq L \text{ and } \frac{x}{y} = -\tan\phi \\ 0 & \text{otherwise} \end{cases} \tag{2.11}$$

If φ is assumed 0, then the discrete version of equation 2.11 can be obtained approximately;

$$h(i,j) = \begin{cases} \frac{1}{L} & \text{if } n_1 = 0, |n_2| \leq \left\lfloor \frac{L-1}{2} \right\rfloor \\ \frac{1}{L}\left\{(L-1) - 2\left\lfloor \frac{L-1}{2} \right\rfloor\right\} & \text{if } n_1 = 0, |n_2| = \left\lceil \frac{L-1}{2} \right\rceil \\ 0 & \text{otherwise} \end{cases} \tag{2.12}$$

2.1.2 Rotational Blur model

When there is a rotational motion between a scene and the camera over the period of exposure $T = [0, t_{exposure}]$, rotation motion blurred images are recorded. Unfortunately, homogeneous blurring assumption is not valid here because blurring paths of rotation motion blurred image are circular arcs and the blurring extents are varied with the radius r. Therefore, rotational movements caused to spattially-variant blur on images.

Rotation motion blurred image can be formulated as in the work of Wang Wenying et al. [61];

$$z = \frac{1}{T}\int_0^T u\left(x - x_0(t), y_0(t)\right) dt. \tag{2.13}$$

where $x_0(t) = r\cos(\omega t)$, $y_0(t) = r\sin(\omega t)$, $r = \sqrt{x^2 + y^2}$ and ω is the angular velocity

of rotation. If image plane is represented in polar coordinate system, then rotation motion blurred image can be formulated as in equation 2.14;

$$z(r,\theta) = \frac{1}{T}\int_0^T u(r,\theta-\omega t)\,dt. \tag{2.14}$$

where (r,θ) is the polar coordinate of image point. Let's change the parameters of equation 2.14 as $l = r\theta$ $s = r\omega t$, $N_r = r\omega T$ and suffix r;

$$z_r(l) = \frac{1}{N_r}\int_0^{N_r} u_r(l-s)\,ds. \tag{2.15}$$

The discrete version of equation 2.15;

$$z_r(i) = \frac{1}{N_r}\left[u_r(i) + u_r(i-1) + \ldots + u_r(i-N_r+1)\right]. \tag{2.16}$$

where $i = 1, 2, \ldots, M_r$ and M_r is the period of pixels sequences in the blurring circular arc. Then PSF in discrete form can be written as

$$h_r(i) = \begin{cases} \frac{1}{N_r} & \text{if } 1 \le i \le N_r \\ 0 & \text{if } N_r < i \le M_r \end{cases} \tag{2.17}$$

Then if equation 2.16 is revisited for simplification;

$$z_r(i) = \sum_{m=0}^{N_r} u_r(m)\,h_r(m) = u_r(i) * h_r(i). \tag{2.18}$$

where $h_r(i)$ is PSF which blurs the image rotationally, $N_r = r\omega T$ is the blur intensity and $\theta = \omega T$ shows angle of blurring during the exposure time T. There is a correlation between blur intensity N_r and radius r, therefore, the blurring intensity can be identified separately with different radius. The intensity of blur can be obtained by calculating the blur angle due to the fact that θ is constant when r is changing. After all, the PSF is obtained when the blur intensity is identified.

As it is mentioned above, the intensity of blur changes with radius r.Therefore, it is important to find the center of rotation and then to obtain the pixels along circular blurring arcs. For

example, center of the blur is taken as the center of image in the work of Wang Wenying et al. [61] and then the idea of Bresenham's circle generation algorithm is used to obtain the pixels on the circular arcs.

Another example is the work of Georg Klein and Tom Drummond [30], it is again assumed that there is no translation and camera rotates with constant angular velocity about a center of rotation. It has two steps that determine the axis of rotation and then the intensity of blur. All points in the image are blurred rotationally on circular arcs which have common center and there is no blur function at the perpendicular direction towards the center of blurring circles. Therefore, edges of image that arise radially from center on the blurring arcs are degraded by blur, while edges lays tangentially from center on the arcs are preserved. In the first step this observation is taken into account so edge detector is used to find the center of blurring circles which is the most perpendicular to all remaining edges of the blurred image. At the second step, blur intensity is calculated with a sign ambiguity in blur magnitude by using the observation that the blur length cannot exceed the length of the shortest intensity ramp if the first step gives the center of the blur correctly and the samples are therefore taken along the direction of blur.

2.1.3 Radial Blur model

Radial blur sometimes referred to as a zoom blur is a type of spatially-variant motion blur which occurs while a visual sensor is moving rapidly towards an object of interest during the acquisition process. In this type of blurring the apparent motion is different at each point in the object. In other words, there is an increasing blur while moving outward from the center of the image. Figure 2.2 demonstrates the model of the blur and how blur smears are directed along radial lines. While camera is coming closer to the object, camera's field of view triangle is getting smaller however there is no change on the size of the object as it is shown on the left side of the Figure 2.2. If the exposure time of camera is significant comparatively to the relative velocity and distance between camera and object of interest, then the image will be radially blurred due to the time-dependent scaling factor of the object in the camera image plane. Aerial photography and video-based missile systems can be given as example situation where this problem may occur.

There are only few works on radial blurred images in the literature [60], [9]. The mathematical

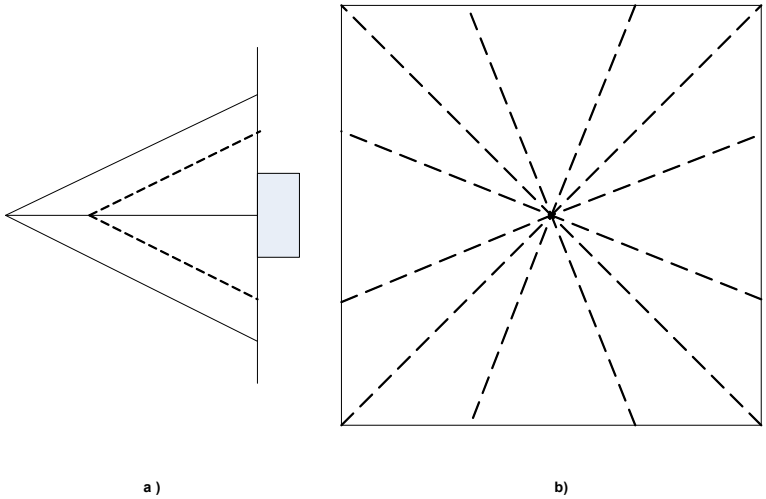


Figure 2.2: The model of Radial blur is given at **a** and how radial lines smears from blur center is given at **b**.

model proposed by Webster and Reeves [60] which is based on transformation of a spatially-invariant blur into a spatially-variant system on a new coordinate system is given here as the model of radial blur 2.19;

$$z_\theta(\phi) = \int_0^T u_\theta(r(\phi, t))\, dt. \tag{2.19}$$

The blur turns out to be one dimensional for any given θ, therefore the equation 2.19 is given with a subscript θ. Where u_θ is the sharp image, z_θ is the radially blurred image, r is the object-plane radial coordinate, θ is the angular coordinate in both the object plane and sensor plane which shows the angle of of every position r, ϕ is taken to be the viewing angle related with a particular pixel in the sensor which has similar function in the sensor plane with r in the object plane and t represents the time.

$$r = x\tan(\phi), \ \frac{dx}{dt} = -v. \tag{2.20}$$

where x is the horizontal spatial coordinate which also shows the distance between camera and object of the interest and v is the camera speed. By applying a change of variables on equation 2.19

$$z_\theta(\phi) = \frac{1}{v}\int_{x_0-vT}^{x_0} u_\theta(x\tan(\phi))\,dx. \tag{2.21}$$

Here, it is aimed to obtain spatially invariant blur model which can be called as a convolution model. The limits of integration is will be changed with logarithmic terms and integrand will contain an exponential term due to achieve this aim.

$$c = \log(x) + \log(\tan(\phi)),\ e^c = x\tan(\phi). \tag{2.22}$$

By differentiating equation (2.22)

$$\begin{aligned} dz = \frac{1}{x}dx = \frac{\tan(\phi)}{e^c}dx \\ dx = \frac{1}{\tan(\phi)}e^c dc \end{aligned} \tag{2.23}$$

If equations (2.22), (2.23) are combined with the equation (2.21);

$$\tilde{z}_\theta(\alpha) = \int_{\rho_1+\alpha}^{\rho_2+\alpha} m_\theta(c)\,dc. \tag{2.24}$$

where

$$\begin{aligned} m_\theta(c) &= e^c u_\theta(e^c),\ \rho_i = \log(d_i),\ \alpha = \log(\tan(\phi)) \\ d_1 &= x_0 - vT,\ d_2 = x_0 \end{aligned}$$

For simplification it can be written as multiplying the integrand by a pulse function;

$$\begin{aligned} \tilde{z}_\theta(\alpha) &= \int_{-\infty}^{\infty} \left[\delta(c-(\rho_1+\alpha))\,\delta(c-(\rho_2+\alpha))\right] m_\theta(c)\,dc \\ &= \left[\delta(c-\alpha-\rho_1)\,\delta(c-\alpha-\rho_2)\right] * m_\theta(\alpha) \end{aligned} \tag{2.25}$$

Blur operation of the work by Webster [60] can be explained in discrete form by explaining their radial sampling method. Radial sampling can be divided into two steps. Firstly, lines of pixels from the center of image to the boundary of image are determined with a certain angle difference consecutively. As a second step, samples are taken by annular region method which has an increasing sampling density from inner parts to outer parts of image on each line and each radial line of points corresponds a column to build a radial blurred image.

2.2 Motion Blur Identification

If the PSF of the motion blur is not given a priori in image deblurring process, the first step is to model the motion blur. Once motion blur is modeled, the parameters of motion blur model must be obtained. This step is called as motion blur identification. If the blur model is chosen as linear uniform motion blur, then the blur identification is to estimate the length and the direction of the motion, for example. Therefore, motion blur identification is a crucial step for deblurring algorithms. There has been many methods proposed for motion blur identification in the image processing community. These methods can be discussed in 3 groups as single image based, multiple images based and external measurements based methods.

2.2.1 Based on single image features

If the relative motion between the camera and the scene is known exactly, the PSF can be calculated analytically. However, extra sensors are necessary to achieve this. Estimating the blur by using the captured image itself is still the most common scenario in literature although it is really a hard problem.

By using single image without any extra knowledge on the motion is called as single image blind deconvolution and it is the most ill-posed hardest type among motion deblur methods due to the fact that there are more unknowns than knowns. In spite of being an ill-posed problem, it has been studied well in the literature. In fact, it is very common problem especially in digital photography due to the camera shake and low lighting conditions which cause long exposure time in consumer cameras. To overcome the difficulties of this problem, limited motion types are chosen to constrain the problem or statistical methods are used as an auxiliary knowledge.

In traditional methods PSF is generally modeled as a low pass filter caused by uniform motion and these methods fail for intensive blurred image case. However, the work of Fergus et al. [19] showed that his method can work under intensively and complex blur. That is why it is one of the most successful work done with a single image. It depends on the fact that natural scene images have certain distributions of image gradients. Therefore heavy-tailed distributions in the gradients of natural images are used while unblurred image and blur kernel are estimated. But even in this work some manual inputs are necessary.

Recent work of Shan et al. [52] model motion blur as maximum a posteriori (MAP) problem. It tries to estimate the parameters of PSF by iterative optimization which may even start from a rough initial kernel estimate. This method has three main contributions. By using a new image noise model, the errors caused by image noise estimation and errors caused by blur kernel estimation is separated. While the parameters of PSF is being refined, a new smoothness constraint is used to suppress ringing artifacts. Finally, computationally hard steps of optimization algorithm is moved to be done in the frequency domain.

Another motion blur identification method from a single image is proposed by using the α-motion blur constraint model, [17]. This method depends on digital matting [35] is the process of extracting a foreground object from an image along with an opacity estimate for each pixel covered by the object. It is derived from the assumption that there is a linear constraint between the image derivatives of the α-channel, the motion blur parameters, and a simple binary free parameter (i.e., +1 or -1). Therefore, α-channel of the image is considered instead of working on blurred image.

These methods are mostly used for motion blur in digital photography because there is no chance to capture multiple images or use external sensors in small high resolution cameras. In robotic applications, multiple images based methods and usage of external sensors are more common. Similarly, we also concentrate on video sequences captured by our experimental robotic platform.

2.2.2 Based on multiple images

As it is mentioned in the Section 2.2.1, blind deconvolution approaches should be used if the relative motion between camera and the scene is not given. One method to cope with

this significantly challenging problem is to use multiple images. Moreover, more complex motion models than it can be considered in the methods based on single image can be and are considered in the methods based on multiple images by using extra information coming from additional images.

An early example of this idea is proposed by Bascle et al. [5] which takes a blurred image sequence as input and gives a single unblurred high-resolution image as output. In Bascle's work, it is assumed that the motion is stable and does not change frame to frame which means that direction of the motion blur is the same in all frames. Therefore, identification of motion blur can be done by motion analysis of frame sequence such as motion blur direction is the common estimated motion direction and the blur intensity is proportional to the common estimated motion magnitude.

However, motion blur direction can be different from one frame to other frame in reality especially in robotic applications. Rav-Acha and Peleg [49] work on two blurred images which have different blur direction (perpendicular to each other). A Gaussian pyramid which goes from the smoothed and sub-sampled images to the high-resolution images is used to estimate the blur parameters for each images.

Another interesting work among the multiple image based methods is proposed by Cho et al. [15] to deal with spatially variant motion blur. Sequential frames from a video is used for deblurring and first, each frame is segmented into small regions in which motion can be assumed as uniform. Then PSFs' identifaction for each region is done by energy minimization approach.

The most interesting work for our research is the work by Agrawal [3] which can be counted among the multiple image based methods, although it also has control on the exposure time of the camera. Instead of recording the video with a constant exposure time, exposure time is being changed frame to frame according to determined exposure time sequence. This gives frames blurred in the same direction but in different magnitudes which is called invertible PSFs because the zeros of PSFs in frequency domain are eliminated by combining other PSFs from frame sequence recorded in different exposure times. This exposure time sequence is repeated while recording the video and the frames which has same exposure time are matched to identify the motion blur parameters.

Using multiple images or video sequences supply us more information about motion and it may even turn the ill-posed problem into a well-posed problem. Even if it is applied for 1D motion, there is even an approach among the multiple image based algorithms to cope with the non-invertible corruption by obtaining invertible PSFs [3]. We are planning to improve this approach for more complex motion cases and then to integrate this approach into our robotic application to achieve a much more robust vision sub-system in the presence of legged motion and the resulting visual disturbances.

2.2.3 Based on external measurements

As it is mentioned at the beginning of Section 2.2.1, the restoration problem will be much easier if the motion is known as a priori knowledge. That is why, there are works which try to use inertial sensors for accurate motion knowledge. Inertial sensor such as an Inertial Measurement Unit (IMU) is robust to even sudden and large motions and can work at higher frequencies than usual cameras. The weak part of these sensors is the accumulative error during integration time. Therefore they are not capable of identification and deblurring without image data.

The work of Klein and Drummond uses gyroscope and camera data together to increase the performance of their parametric edge detection algorithm. Although their aim is to obtain better tracking performance, motion parameters as a motion matrix is estimated by using combined gyroscope and camera data. Moreover, the improvement in the edge detection step of the algorithm that they achieved by using motion estimation is one of the motivation point for us to analyze feature detectors under motion blur.

CHAPTER 3

IMAGE FEATURE DETECTORS FOR ROBOTIC APPLICATIONS

3.1 Importance of Feature Detectors For Robotic Applications

A local feature is an image pattern which differs from its immediate neighborhood. It is usually associated with a change of an image property or several properties simultaneously, although it is not necessarily localized exactly on this change. The image properties commonly considered are intensity, color, and texture. Local features can be points, but also edgels or small image patches. Typically, some measurements are taken from a region centered on a local feature and converted into descriptors.

The descriptors can then be used for various applications in computer vision and robotics. In object recognition, it is important to find a mapping between model and image. Another example can be given as navigation algorithms in mobile robotics. Most real-world environments provide salient features which are useful for navigation and can be extracted from vision data. It turns out that features with a simple geometry are a good choice as they are relatively easy to obtain and of frequent occurrence in man-made environments.

Mobility requires the knowledge of one's own position in the environment. Feature based algorithms gain have important role for localizing a robot precisely and robustly particularly when continuously updating the robot's pose during motion in real-time. Mobility requires further a map of the environment in which one is supposed to navigate. Extracting features from sensory data is crucial that the robot can start to build a map of a previously unknown environment autonomously.

3.2 Harris Corner Detector

The basic idea of the corner detection is to find points where two edges meet. In other words, it aims to find high gradient in two directions. And one of the most popular corner detector algorithm is Harris corner detector because it is reasonably invariant to rotation, different sampling and quantization, small changes of scale and small affine transformations, illumination variation and image noise [51]. It is used in many computer vision applications such as matching, finding correspondence, tracking and so on. Therefore, The derivation of the Harris corner detector [25] is presented in this part of the report.

The Harris corner detector is based on the local auto-correlation function of a signal. The local auto-correlation function captures the structure of the local neighborhood by measuring the changes of the signal with patches shifted by a small amount in all directions. A discrete predecessor of the Harris detector was proposed by Moravec [44].

The auto-correlation function can be written as;

$$c(x,y) = \sum_{W} \left[I(x_i, y_i) - I(x_i + \Delta x, y_i + \Delta y) \right]^2 \tag{3.1}$$

where I denotes the image function, (x_i,y_i) are the points in the window W centered on (x,y) and a shift(Δx,Δy).

If the shifted function is approximated by the first-order Taylor expansion;

$$I(x_i + y_i + \Delta y) \approx I(x_i, y_i) + I_x(x_i, y_i)\,\Delta x + I_y(x_i, y_i)\,\Delta y = I(x_i, y_i) + \left[I_x(x_i, y_i)\, I_y(x_i, y_i) \right] \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \tag{3.2}$$

where I_x, I_y are partial derivatives of I(x,y).

If we substitute approximation equation (3.2) into equation (3.1),

$$c(x,y) = \sum_W \left[I(x_i,y_i) - I(x_i+\Delta x, y_i+\Delta y)\right]^2 \quad (3.3)$$

$$= \sum_W \left(I(x_i,y_i) - I(x_i,y_i) - \left[I_x(x_i,y_i)\, I_y(x_i,y_i)\right] \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \right)^2 \quad (3.4)$$

$$= \sum_W \left(-\left[I_x(x_i,y_i)\, I_y(x_i,y_i)\right] \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \right)^2 \quad (3.5)$$

$$= \sum_W \left(\left[I_x(x_i,y_i)\, I_y(x_i,y_i)\right] \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \right)^2 \quad (3.6)$$

$$= [\Delta x \Delta y] \begin{bmatrix} \sum_W (I_x(x_i,y_i))^2 & \sum_W \left(I_x(x_i,y_i)\, I_y(x_i,y_i)\right) \\ \sum_W \left(I_x(x_i,y_i)\, I_y(x_i,y_i)\right) & \sum_W \left(I_y(x_i,y_i)\right)^2 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \quad (3.7)$$

$$= [\Delta x \Delta y]\, Q(x,y) \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \quad (3.8)$$

where matrix Q(x,y) captures the intensity structure of the local neighborhood. Eigenvalues of matrix Q(x, y) gives us a measure such as;

- If the local auto-correlation function is flat which shows that both eigenvalues are small, then the windowed image region is of approximately constant intensity.
- If the local auto-correlation function is ridge shaped which shows that there is one strong eigenvalue , then this indicates an edge.
- If the local auto-correlation function is sharply peaked which shows that both eigenvalues are strong, then this indicates a corner.

In this report, the Harris corner code by Peter Kovesi [32] is utilized as the Harris Corner code. The flow of the algorithm can be visualized in Fig. 3.1. For each frame, the algorithm is run and the 18 corner positions on the image are found.

3.3 Canny Edge Detector

Edges in images are areas with strong intensity contrasts from one pixel to the next which shows that outlines of an object and boundaries between objects and the background in the

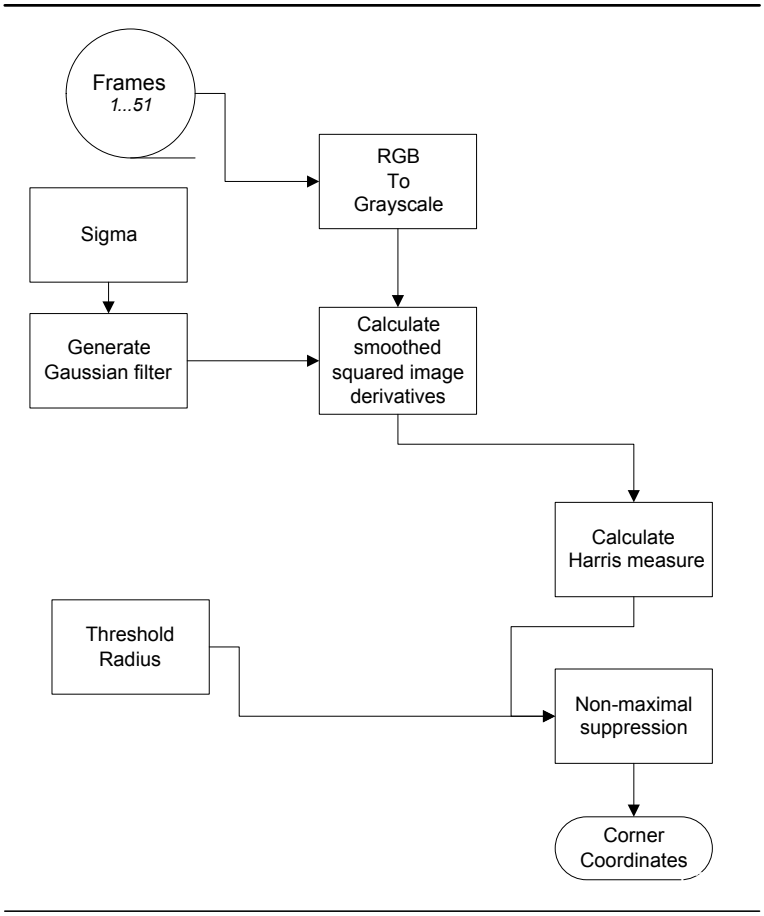


Figure 3.1: Flowchart of Harris corner detector algorithm

image. Edge detection is a fundamental tool used in most image processing applications as a preprocessing step to feature extraction and object segmentation.

The Canny algorithm is one of the most commonly used image processing tools, detecting edges in a very robust manner and using an optimal detector which is based on finding as many real edges in the image as possible by minimizing the error rate, marking edges as close as possible to the edge in the real image to maximize localization, and marking edges only once without false edges created by image noise.

Canny's work is based on expressing the preceding criterias mathematically and then find optimal solutions to these formulations. Using numerical optimization with 1-D step edges corrupted by additive white Gaussian noise led to the conclusion that a good approximation to the optimal step edge detector is the first derivative of a Gaussian:

$$\frac{d}{dx}e^{\frac{-x}{2\sigma^2}} = \frac{-x}{\sigma^2}e^{\frac{-x^2}{2\sigma^2}} \tag{3.9}$$

1-D approach works by applying in the direction of the edge normal. However, the direction of the normal is not known beforehand in 2-D. Therefore, 1-D edge detector should be applied in all possible directions while it is being generalized to 2-D. Smoothing the image with a circular 2-D Gaussian function, computing the gradient of the result and then using the gradient magnitude and direction is a good approximation for this.

Let $I^n(x,y)$ denote the nth frame of input video and $G(x,y)$ denote the Gaussian function;

$$G(x,y) = e^{-\frac{(x^2+y^2)}{2\sigma^2}} \tag{3.10}$$

Smoothed image $I_s^n(x,y)$ is obtained by convolving G and I^n;

$$I_s^n(x,y) = G(x,y) * I^n(x,y) \tag{3.11}$$

This step is followed by computing the gradient magnitude and direction;

$$g_x = \frac{\partial I^n}{\partial x} = I^n(x+1,y) - I^n(x,y) \tag{3.12}$$

$$g_y = \frac{\partial I^n}{\partial y} = I^n(x, y+1) - I^n(x, y) \tag{3.13}$$

$$M(x, y) = \sqrt{g_x^2 + g_y^2} \tag{3.14}$$

and

$$\alpha(x, y) = \arctan\left[\frac{g_x}{g_y}\right] \tag{3.15}$$

$M(x, y)$ contains wide ridges around local maxima because it is generated by using the gradient. Therefore the next step is using nonmaxima suppression to thin those ridges. It can be formulated as;

- Find the direction of all possible edge direction d_k that is closest to $\alpha(x, y)$ in the given region.
- If the value of $M(x, y)$ is less than at least one of its neighbors along d_k, then $g_N(x, y) = 0$ which means that it is suppressed otherwise $g_N(x, y) = M(x, y)$.

After obtaining nonmaxima-suppressed image $g_N(x, y)$, the final step is to threshold $g_N(x, y)$ to reduce false edge points. Canny's algorithm uses hysteresis thresholding which contains two thresholds such as upper threshold T_H and a lower threshold T_L. Thresholding step can be formulated as

$$g_{NH}(x, y) = g_N(x, y) \geq T_H \tag{3.16}$$

and

$$g_{NL}(x, y) = g_N(x, y) \geq T_L \tag{3.17}$$

Due to the fact that g_{NL} contains all the nonzero pixels g_{NH}, it should be eliminated;

$$g_{NL}(x, y) = g_{NL}(x, y) - g_{NH}(x, y) \tag{3.18}$$

The nonzero pixels in $g_{NH}(x,y)$ and $g_{NL}(x,y)$ are called as strong and weak edges, respectively. After the thresholding step, all strong pixels in $g_{NH}(x,y)$ are marked as valid edge pixels. Depending on the value of upper threshold T_H, the edges in $g_{NH}(x,y)$ typically have gaps. Longer edges are formed using the following procedure:

1. Locate the next unvisited edge pixel, p, in $g_{NH}(x,y)$.
2. Mark as valid edge pixels all the weak pixels in $g_{NL}(x,y)$ that are connected to p using 8-connectivity.
3. If all nonzero pixels in $g_{NH}(x,y)$ have been visited go to Step 4. Else return to first step.
4. Set to zero all pixels in $g_{NL}(x,y)$ that were not marked as valid edge pixels.

In this report, the Canny edge code by Peter Kovesi [32] is utilized as the Canny Corner code. The flow of the algorithm can be visualized in Fig. 3.2. For each frame, the algorithm is run and the edges on the checkerboard plate are trying to obtain.In summary,the Canny edge detection algorithm works in a multi-stage process;

- Smoothing the image and eliminating the noise by convolving with a Gaussian filter.
- Finding the gradient magnitude and orientation using finite-difference approximations for the partial derivatives.
- Applying non-maximal suppression to the gradient magnitude for finding the local maxima in the direction of the gradient.
- Using hysteresis thresholding algorithm which has some adaptivity to the local content of the image to detect and link edges.

3.4 SIFT: Scale Invariant Feature Transform

In general a good local feature should have these key properties;

- It should be easy to extract.

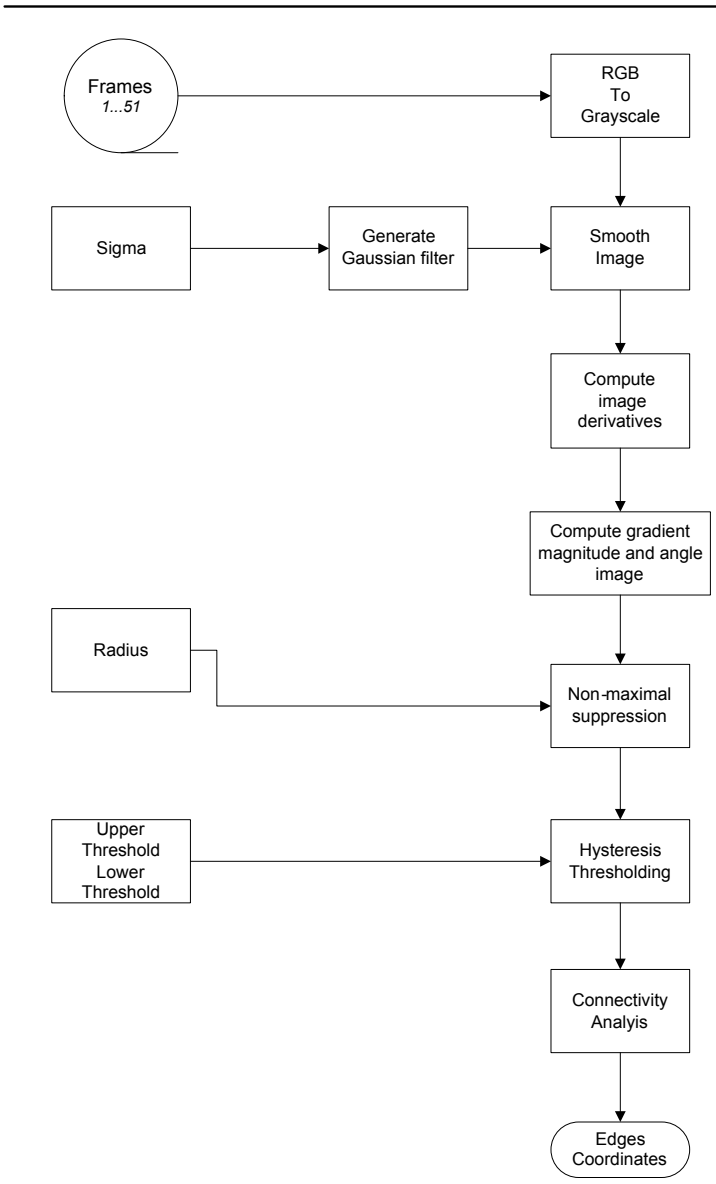


Figure 3.2: Flowchart of Canny edge detector algorithm

- It should be easy to match.
- It should be robust to image noise.
- It should be robust to illumination changes.
- It should be robust to minor changes in viewpoint.
- It should be independent from rotation and scaling.

Scale Invariant Feature Transform (SIFT) developed by David Lowe [38] is a powerful image feature detection and extraction method which claims to satisfy most of these properties such as independence from scale and rotation as well as being robust to affine distortions viewpoint changes, noise and illumination changes. Besides, being highly distinctive, SIFT's features can be correctly matched against a large database of features. In view of these favorable properties, SIFT has gained huge popularity in many areas, such as object recognition, stereo matching, 3D structure estimation and motion tracking. Therefore, it is also an important approach for robot vision applications.

SIFT algorithm takes gray-scale input images and the main steps of detection for SIFT feature;

- Interest points candidates are found by scale-space extrema detection
- Location, scale and contrast is calculated for each candidate keypoints and unstable keypoints are discarded.
- Orientation assignment for each keypoint location are done by calculating local gradient directions.
- Keypoint descriptors are constructed by using the local gradients around each keypoint at the selected scale.

First of all,local extrema of difference-of-Gaussian filters at different scales are the interest points for SIFT algorithm. Therefore, candidate keypoints which are stable against scale change are to be detected by searching among all possible scales.

Variable scale Gaussian function given in equation 3.19 is used as scale space kernel.

$$G(x, y, \sigma) = 1/\left(2\pi\sigma^2\right) e^{-\left(x^2+y^2\right)/\sigma^2}. \tag{3.19}$$

By using equation 3.19, the scale space of an imput image $I(x, y)$ can be obtained as;

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y). \tag{3.20}$$

Difference-of-Gaussian function which is the difference between two images at scales $k\sigma$ and σ is used for detection of keypoints locations and it is given by

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) && (3.21) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) && (3.22) \end{aligned}$$

Difference of Gaussian (DoG) images from adjacent images are generated as the first step of SIFT algorithm towards the detection of interest points. The difference of Gaussian function is chosen because it is efficient function to compute and it provides an approximation to the scale-normalized Laplacian of Gaussian $\sigma^2\nabla^2 G$ which gives the most stable scale invariant image features. Figure 3.3 shows the computation of the difference-of-Gaussian function.

As it is seen in figure 3.3, the input image is convolved by Gaussians and the scale space images are obtained. These scale space images are grouped by octaves. The scale of each image in each octave differs from the scale of previous image by a constant factor k. After the value of k is choosen, a fixed number of s intervals per octave is obtained. The relationship between k and s is given as;

$$k = 2^{1/s}. \tag{3.23}$$

Therefore, the last image in each octave has twice the scale of the first image of each octave. s+3 images are created for each octave to be able to find local-extrema through a complete octave. As it is shown in the right side of figure 3.3, the difference-of-Gaussian images are obtained by taking the difference of adjacent blurred image in each octave. The first image of the following octave is obtained by downsampling by 2 the last image of the previous octave.

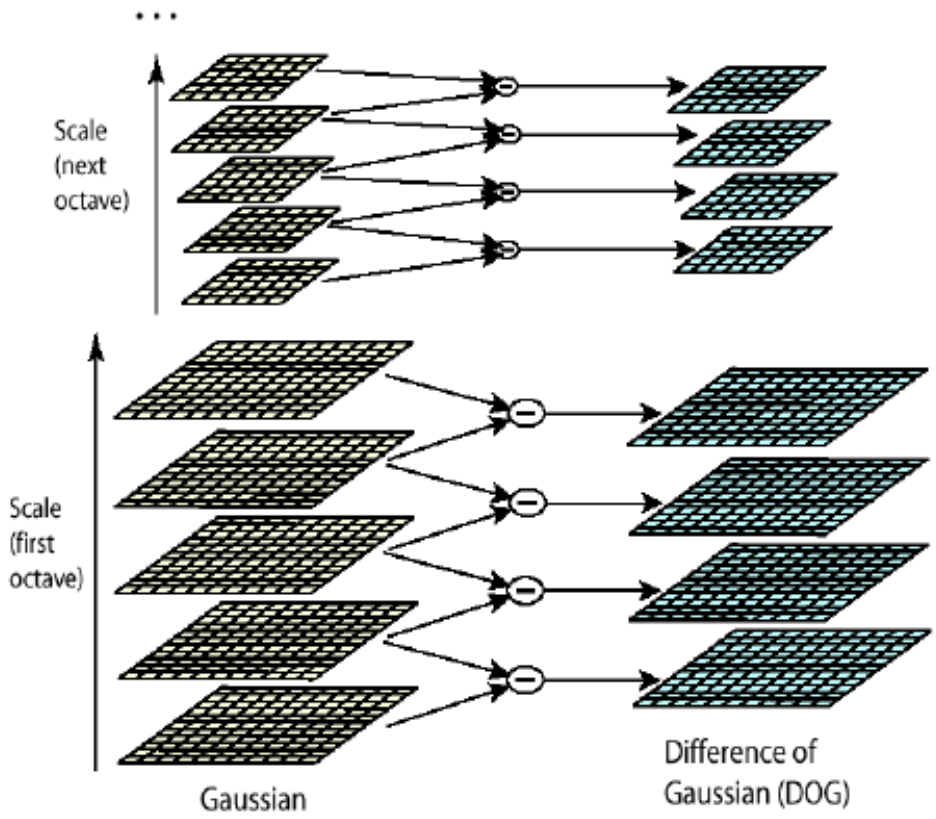


Figure 3.3: Figure shows that the scale space images which are obtained by convolving the input image by Gaussians with different scale factors and the computation of the difference-of-Gaussian images [39].

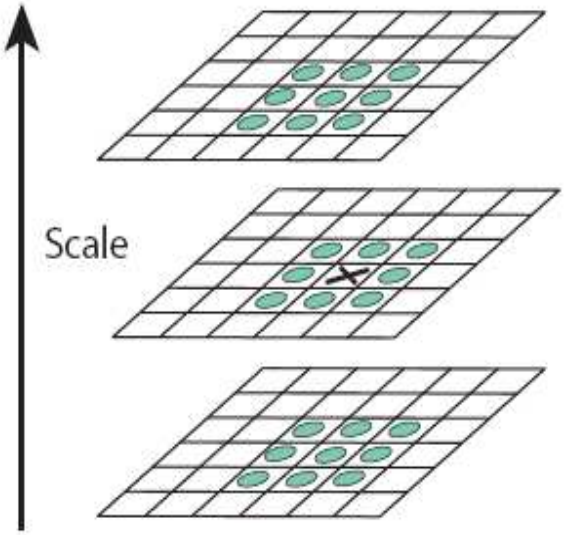


Figure 3.4: Each pixel marked with ”x” with its 26 neighbors in $3 \times 3 \times 3$ neighborhood which consists 8 on the same scale image, 9 on the scale above image and the 9 on the scale below image [39].

Keypoints of SIFT are obtained by searching for the local maxima and minima of the DoG function. Local minima and maxima detection is shown on figure 3.4. At this step, each pixel is compared with its 8 neighbors on the same scale image, plus 9 corresponding neighbors on the one scale above image and one scale below image. The pixel is selected as a candidate keypoint if it is local maximum or minimum point.

After determining candidate keypoints, a 3D quadratic function is fit to the nearby pixels of the candidate keypoints to increase the stability [41]. By this step;

- Low contrast keypoints are eliminated.
- Keypoints chosen among the edge points are eliminated.
- Orientation of keypoints are calculated in sub-pixel accuracy.

The Taylor series expansion of the DoG function is used in this approach [41]. If the origin of it is shifted to the sample point;

$$D(x) = D + \frac{\partial D^T}{\partial x} + \frac{1}{2} x^T \frac{\partial^2 D}{\partial x^2} x. \tag{3.24}$$

In equation 3.24, D and its derivatives are calculated at the sample point and $x = (x, y, \sigma)^T$ is the offset from this point. The derivative of this equation (3.24) with respect to x and equating it to zero gives the location of extrema point $\hat{x}$;

$$\hat{x} = -\frac{\partial^2 D^{-1}}{\partial x^2} \frac{\partial D}{\partial x}. \tag{3.25}$$

If the offset of the accurate keypoint location from the sample point location $\hat{x}$ is greater than 0.5 in any direction; the extremum which means that accurate location is closer to another pixel. In such a case, the sample point is changed and the same procedure is repeated for the new one. Sub-pixel localization is satisfied as adding the final offset $\hat{x}$ value to the location of the sample point.

Another aim of this step is to eliminate keypoints with low contrast in their neighborhood. This can be achieved by substituting equation (3.25) into equation (3.24). It increase the stability of keypoints and mathematically results as;

$$D(\hat{x}) = D + \frac{1}{2} \frac{\partial D^T}{\partial x} \hat{x}. \tag{3.26}$$

If the value of $D(\hat{x})$ for a candidate keypoint is less than 0.03 (pixel values are assumed to be in the range $[0, 1]$), this point is eliminated.

Another stability problem is caused by the keypoints chosen along the edges. These points are mostly unstable and sensitive to noise; therefore, it will be mentioned how these points are removed in this step. The principle curvatures of DoG function at stable points are comparably large in both directions; however, the DoG function at points lying along edges directions has small principle curvatures along the direction and a large principal curvature in the perpendicular direction.

The principal curvatures can be calculated by using the eigenvalues of the Hessian matrix;

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix} \tag{3.27}$$

The eigenvalues of the Hessian matrix are proportional to the principal curvatures of the DOG function. At this point, the main idea of Harris Corner Detector is used [25]. That is, there is no need to compute eigenvalues explicitly because only the ratio of the eigenvalues is necessary. Let α and β are eigenvalues of Hessian matrix H and the ratio between them given as;

$$\alpha = r\beta. \tag{3.28}$$

If the trace of H and determinant of H is calculated;

$$Tr(H) = D_{xx} + D_{yy} = \alpha + \beta \tag{3.29}$$

$$Det(H) = D_{xx}D_{yy} - \left(D_{xy}\right)^2 = \alpha\beta \tag{3.30}$$

As it is shown in equations 3.29 and 3.30, the trace of Hessian gives us the sum of the eigenvalues of H and the determinant of Hessian matrix gives us the product of the eigenvalues of H. Then;

$$\frac{Tr(H)^2}{Det(H)} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r\beta+\beta)^2}{r\beta^2} = \frac{(r+1)^2}{r}. \tag{3.31}$$

The result of the equation 3.31 depends on the ratio r not the eigenvalues. This ratio has the minimum value when r is equal to 1. In other words, it has the minimum value when the

eigenvalues of H are equal to each other. Therefore, comparing this value against a threshold is sufficient to check the ratio of the principal curves of DoG function instead of calculating each eigenvalue. This threshold is given as 10 in the paper of Lowe [39].

After the stable keypoints are selected with their locations in sub-pixel accuracy, a scale and rotation invariant descriptor should be assigned to each keypoint to characterize it. This descriptor is obtained by computing a gradient orientation histogram in the neighborhood of the keypoint. Here, invariance in scale can be achieved by selecting the Gaussian smoothed image L(x,y) which has the closest scale to the scale of the keypoint, and determining its orientation on this image. Similarly, invariance in rotation can be achieved by assigning an orientation to each keypoint, and computing the keypoint descriptor relative to this orientation.

For each keypoint, these calculated orientations around a keypoint are added up to a histogram which has 36 bins for 360° range. The contribution of each neighboring sample is weighted by the gradient magnitude and a Gaussian function with a scale that is 1.5 times the scale of the keypoint. Peaks of the histogram correspond to dominant orientations. That is, the direction of the histogram maximum, and any other direction within 80% of the maximum value corresponds to a new keypoint. Hence, SIFT may give some keypoints which have the same location at the end. But these have different orientations and also increase the matching stability. For better localization, the last step of orientation assigment is to fit a parabola to the 3 histogram values around the peak.

Figure 3.5 shows how to obtain the feature descriptor. These arrows represent local gradient magnitudes and orientations. Here, the Gaussian image closest in scale to the keypoint's scale gives the orientation data. The gradient magnitudes are weighted by a Gaussian with scale 1.5 times the scale of the keypoint, and added up to a histogram to construct the descriptor. A set of orientation histograms on 4×4 pixel neighborhoods is used to compute the feature descriptor. The gradient samples have affect on both of its adjacent histogram entries which depend on the original orientation of the sample and the central value of the bin 'd'. This weighting factor is taken as (1-d).

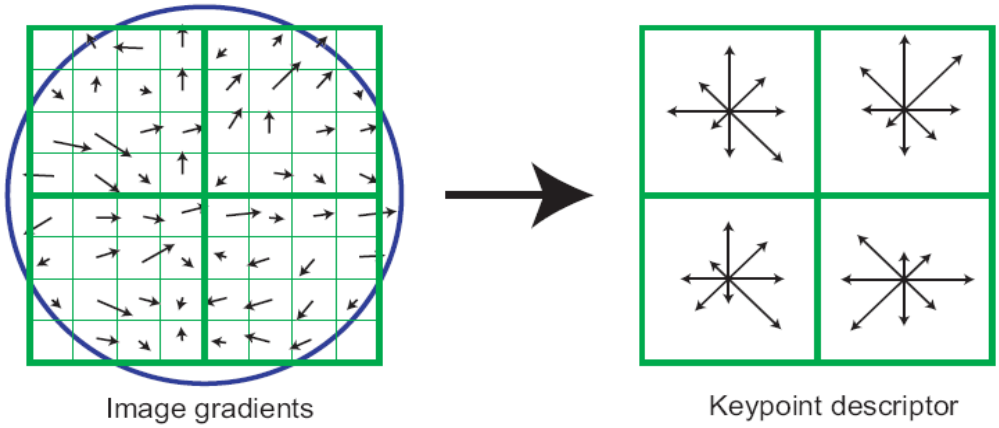


Figure 3.5: The figure on the left shows that gradient magnitudes and orientations of the local image in a 8x8 neighborhood of the keypoint. These are weighted by a Gaussian window, represented by the overlaid circle. The figure on the right shows the magnitudes of histogram entries in 4×4 regions [39].

As it is shown in figure 3.5, histograms have 8 bins each, and each descriptor contains an array of 4 histograms around the keypoint. Therefore, a SIFT feature vector has $4 \times 4 \times 8 = 128$ elements.Each histogram entry represents an element of the descriptor vector. The SIFT code by A. Vedaldi is used as the SIFT code in this thesis.

3.5 Effects of Motion Blur on Feature Detectors

Although there are many feature based algorithms and work related with motion deblurring in computer vision literature, there is no work which investigates the specific effects of motion blur on feature detectors. Therefore, we create an evaluation procedure to characterize feature detectors.

Mainly all feature detectors have three fundamental aims.

- A good feature detector should detect all features without giving fake features. In fact the features found by detector should be the real correct features.
- A good feature detector should assign all features in their real coordinates or as close as possible to their real coordinates. In fact, any feature should be found with minimum distance to real feature.
- A good feature detector should be robust to the image noise. In fact, there should not be multiple features found by detector around a single feature.

The essence of our work is in expressing these criteria and then attempting to evaluate feature detectors according to these under motion blur effect.

CHAPTER 4

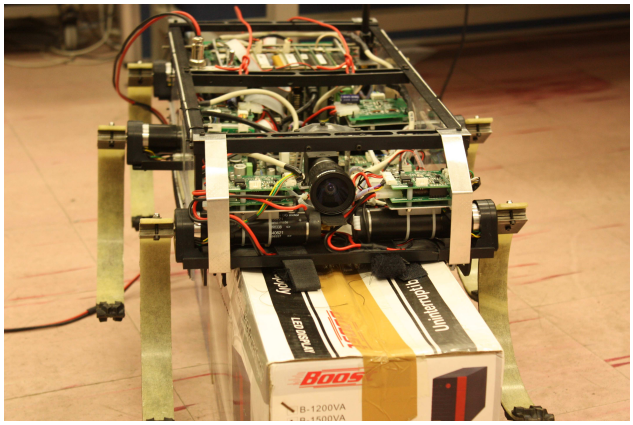
EXPERIMENTAL EVALUATION OF FEATURE DETECTORS UNDER MOTION BLUR

4.1 Experimental Scenario

In our work, PointGrey Flea2 camera is used. As illustrated in Fig. 4.1(a) it can fit into the small, tight spaces. That is why it is commonly preferred in mobile robotics applications. It is mounted on our mobile robotic platform RHex and connected to pc via IEEE 1394b interface as it is shown at Fig. 4.1(b).

All the experiments were held in Robotics and autonomous systems laboratory. A straight route was defined for our robot to walk in different speeds through 3.5 meter path. 3 by 4 checkerboard pattern which is shown at Fig. 4.2 were placed at the end of this 3.5 meter path in the lab environment. This experimental setup can be visualized better by the help of Fig. 4.3.

(a) Flea 2 camera



(b) SensoRHex

Figure 4.1: The Point Grey Flea2 camera is shown with our experimental robotic platform SensoRHex (a) and (b)

Figure 4.2: 3 by 4 Checkerboard plate.

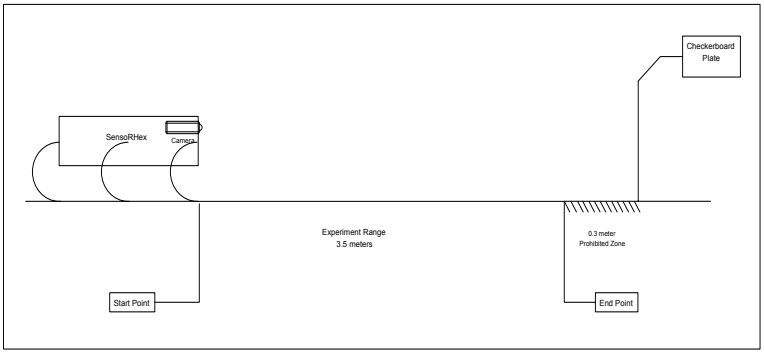


Figure 4.3: Drawing of experimental setup for evaluating feature detectors on motion blurred frames due to legged locomotion.

Video sequences were recorded at a 640 x 480 resolution with a 25 frames per second frame rate in the avi format by using RHex vision library functions on linux platform. Five test video sequences were recorded;

- VSeq 1 was recorded while RHex was standing sti,ll half meter in front of checkerboard plate.
- VSeq 2 was recorded while RHex was approaching to the Checkerboard plate from 4.5 meters with 0.1 velocity coefficient at its slowest walking mode which corresponds to 0.14 m/sec.
- VSeq 3 was recorded while RHex was approaching to the Checkerboard plate from 4.5 meters with 0.4 velocity coefficient at its slow walking mode which corresponds to 0.16 m/sec.
- VSeq 4 was recorded while RHex was approaching to the Checkerboard plate from 4.5 meters with 0.8 velocity coefficient at its fast walking mode which corresponds to 0.25 m/sec.
- VSeq 5 was recorded while RHex was approaching to the Checkerboard plate from 4.5 meters with 1.0 velocity coefficient at its fastest walking mode which corresponds to 0.40 m/sec.

Table 4.1: Speed coefficients of the experimental robotic platform which are used in experiments and the corresponding speed values in m/sec.

Velocity Coefficients	Corresponding Speeds in m/sec
0.1	0.14 m/sec
0.4	0.16 m/sec
0.8	0.25 m/sec
1.0	0.40 m/sec

Fig. 4.4 demonstrates some sample frames from VSeq 4 that were recorded while RHex is running forward and backward to show how the world looks like from RHex's eye. In our experiments, we use forward part of the videos which were recorded while RHex was moving forward. Backward locomotion of the robot degrades the video frames more intensively due to its structure of c shape legs and its walking gate.

Figure 4.4: The image of checkerboard changes through the video frames while RHex is running forward and backward by moderate velocity.

In our work, we investigate performance of harris corner, canny edge detector and sift on the video frames captured by the camera mounted on RHex. The aim of the research is to exploit the effect of legged locomotion on the output of camera, to obtain an idea about possible deblurring methods and to develop an objective performance criteria in motion deblurring by using the motion blur effect on the performance of well known image feature detectors while preparing this experimental scenario.

4.2 Evaluation Criteria

Investigating performance of harris corner, canny edge detector and sift, we need to measure the detection capabilities and the robustness of the methods under motion blur. Intuitively, the evaluation criteria compare the algorithms based on the following behaviors,

- detection rate of interested features on the specified pattern in motion blurred frames. These are classified as ”detected” or ”missed”.
- ability of not generating any false (or spurious) features due to noise or motion blur related artifacts. These are classified as ”false alarm”.
- ability to align the features without any unexpected splitting features. These are classified as ”false split features”.

According to these conditions, performance measures are obtained.

4.3 Experiments with Harris Corner Detector

The first video which was recorded while robot was standing still is used as a reference data. The parameters of algorithm is set and tested according to this video. Later, the performance of harris corner detector algorithm is investigated on other test video sequences respectively.

51 frames and 18 corners in each frame for all 5 video sequences are considered when tests are applied. Firstly, a blur intensity and blur angle is obtained on the most upper-left corner which is marked as the first corner by using equation (4.1) and equation (4.2) for each frame. Let f be the number of frames and c be the number of corner in each frame.

$$\Delta x_{f,c} = \sqrt{\left(x^r_{f,c} - x^e_{f,c}\right)^2} \tag{4.1}$$

$$\phi_{f,c} = \arctan\left(\frac{x^r_{f,c}}{x^e_{f,c}}\right) \tag{4.2}$$

where x^r and x^e denote the real coordinates of corner selected by hand on the frame and the estimated coordinates of corner calculated by harris corner detector algorithm on the frame, respectively. Moreover, equation (4.1) is also used for calculating the error in each detected corner. Fig. 4.5 shows average estimated accuracy errors in corner location of the each frame for the test video sequences. In Fig. 4.5, the peak points which are shown above 15 correspond to frames that are entirely missed in the sense of corner detection. This situation is manifested more clearly in Table 4.2 which shows that total averages of each video.

Vseq 2 is the first and the slowest motion video sequence that we used in our experiments. It was recorded while RHEx was in its walking mode with 0.1 velocity coefficient. In order to see the performance of Harris Corner detector, we have analyzed missed corner rate, error rate of the detected corners, possible false alarms and false splitting effect on this video.

Missed corners appeared in only five frames among 51 frames and only one frame among these five is completely lost which means that none of 18 corners can be detected by algorithm. These results are demonstrated in the Figure 4.6. It shows that corners could be missed only if there is an intensive motion blur.

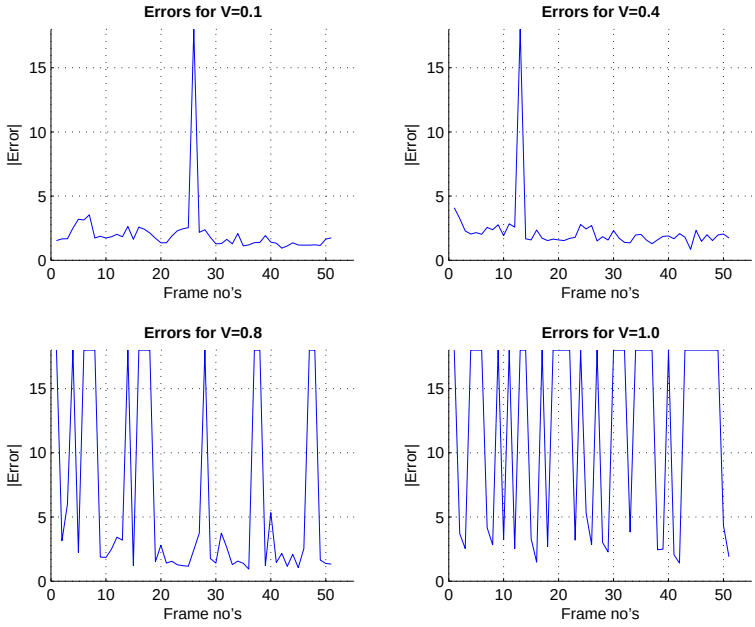


Figure 4.5: Average estimated accuracy errors in corner location for test videos with four test locomotion velocities. The peak points marked above 15 correspond to frames that are entirely missed in the sense of corner detection.

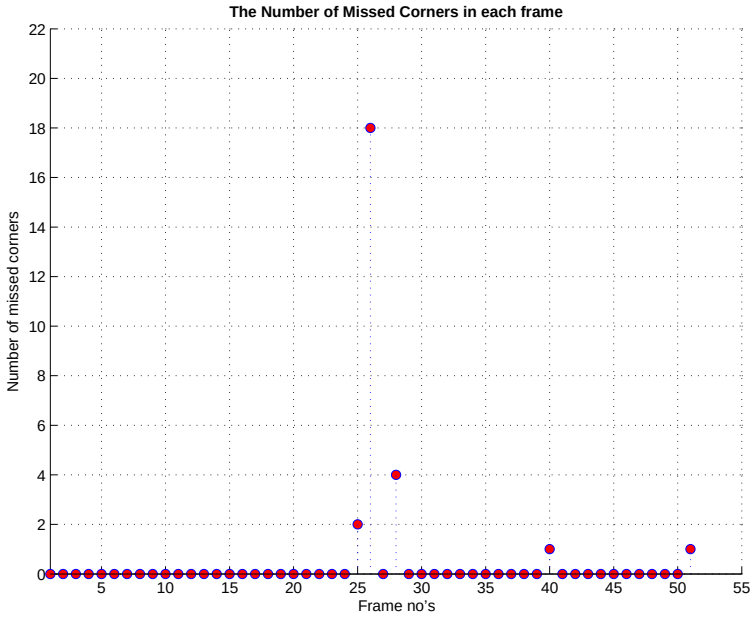


Figure 4.6: The number of missed corners in each frame of the first test video recorded while RHex was walking with 0.1 velocity coefficient at its slowest walking mode.

Table 4.2: Average errors of video sequences.

	E_{ave}	$\tilde{V}$	Number of ”all missed” Frames
Vseq 2	1.7897	0.1	1
Vseq 3	1.9835	0.4	$1/51 \cdot 100$
Vseq 4	2.1216	0.8	$14/51 \cdot 100$
Vseq 5	2.9331	1.0	$30/51 \cdot 100$

Table 4.3: The average rates per corner for Vseq 2

Average Missed Corner rate	5.2000
Average Strabismus rate	4.5385
Average Cross-eyed corner rate	14.2308

The false splitting effect which means that single feature point response can not be achieved was seen in 13 frames among 51 frames in this video sequence. Due to this effect, algorithm finds two or more corners around the true single corner. This effect is mostly seen at the inner corners of pattern and is a good way to parametrize the effect of motion blur even if the frames are not degraded intensively. These results are demonstrated in the figure 4.7. Figure 4.7 shows the number of corners that are degraded by this effect in each frame.

In this video sequence, there is no false alarm which means a fake corner on the plate is not observed. This was achieved due to the carefully fixed threshold value of the algorithm. Besides, the texture of the image is not so complicated as to cause this kind of artificial features.

Vseq 3 is the second motion video sequence that we used in our experiments. It was recorded while RHex was in its walking mode with 0.4 velocity coefficient. In order to see the per-

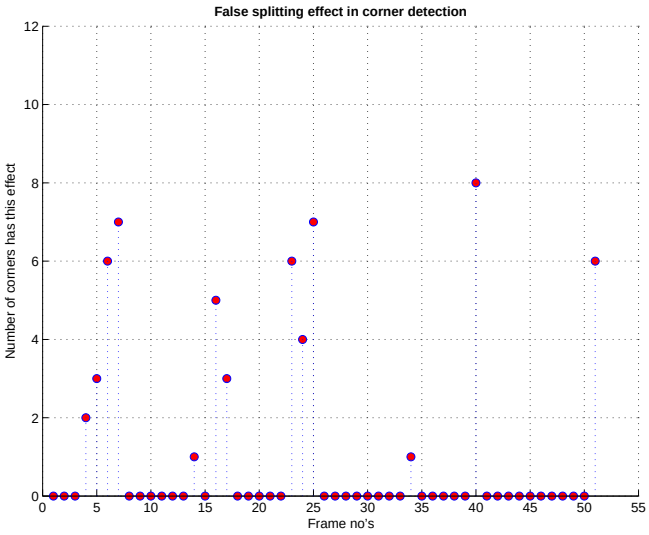


Figure 4.7: The false splitting effect in each frame of the first test video recorded while RHex was walking with 0.1 velocity coefficient at its slowest walking mode.

Table 4.4: The average rates per corner for Vseq 3

Average Missed Corner rate	6.2500
Average Strabismus rate	4.6800
Average Cross-eyed corner rate	14.3200

formance of Harris Corner detector, we have analyzed missed corner rate, error rate of the detected corners, possible false alarms and false splitting effect on this video.

Missed corners appeared in only eight frames among 51 frames and only one frame among this eight is completely lost which means that none of 18 corners can be detected by algorithm. In one more frame among these eight, 16 corners could not be observed. Except for these two frames, the average missed rate is 2.67 in the other six distorted frames. These results are demonstrated in the Figure 4.8. It shows that corners could be missed only if there is an intensive motion blur so this sequence was distorted moderately.

The false splitting effect which reveals that single feature point response can not be achieved was seen in 25 frames among 51 frames of this video sequence. That means half of this sequence is degraded by false splitting effect. Due to this effect, algorithm finds two or more corners around the true corner instead of one. This effect is mostly seen at the inner corners

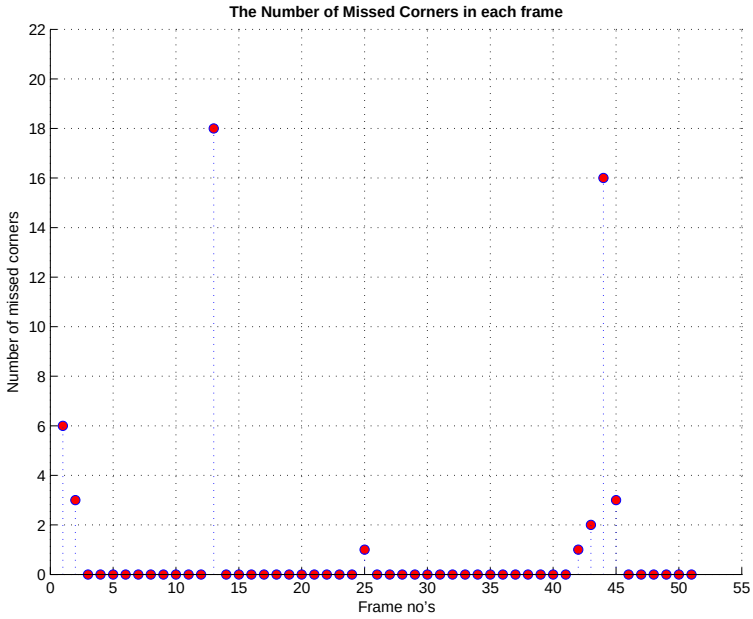


Figure 4.8: The number of missed corners in each frame of the second test video recorded while RHex was walking with 0.4 velocity coefficient at its slow walking mode.

of pattern and is a good way to parametrize the effect of motion blur even if the frames is not degraded intensively. These results are demonstrated in the Figure 4.9. Figure 4.9 shows the number of corners that are degraded by this effect in each frame.

In this video sequence, there is no false alarm which means a fake corner on the plate is not observed. This is because of the carefully fixed threshold value of the algorithm. And also the texture of the image is not so complicated to cause this kind of artificial features.

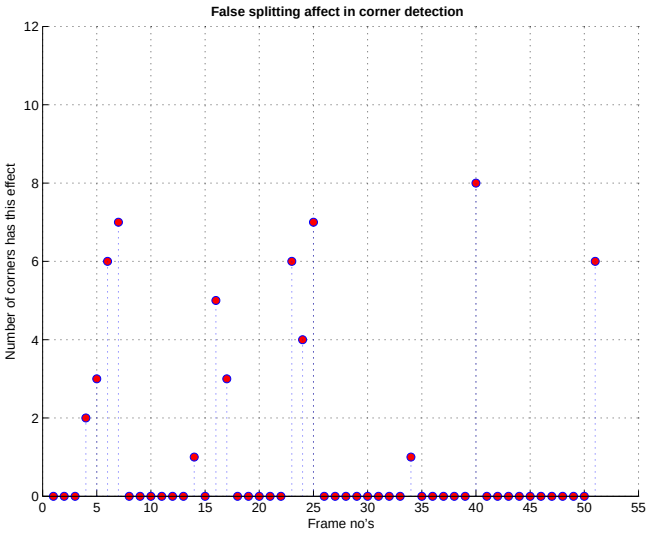


Figure 4.9: The false splitting effect in each frame of the second test video recorded while RHex was walking with 0.4 velocity coefficient at its slow walking mode.

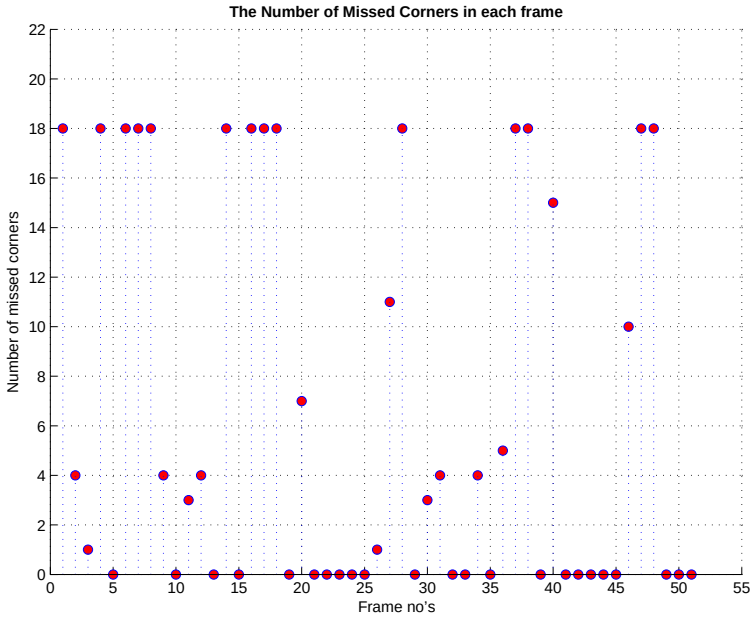


Figure 4.10: The number of missed corners in each frame of the third test video recorded while RHex was walking with 0.8 velocity coefficient at its fast walking mode.

Vseq 4 is the third motion video sequence that we used in our experiments. It was recorded while RHex was in its walking mode with 0.8 velocity coefficient. In order to see the performance of Harris Corner detector, we have analyzed missed corner rate, error rate of the detected corners, possible false alarms and false splitting effect on this video.

Missed corners appeared in only 28 frames among 51 frames and 14 frames among this 28 are completely lost which means that none of 18 corners can be detected by algorithm. Except for these 14 frames, the average missed rate is 5.43 in the other distorted frames. These results are demonstrated in the figure 4.10. It shows that these video sequence is distorted intensively.

Table 4.5: The average rates per corner for Vseq 4

Average Missed Corner rate	11.7143
Average Strabismus rate	4.5556
Average Cross-eyed corner rate	12.3333

The false splitting effect which means that single feature point response can not be achieved was seen in 18 frames among 51 frames in this video sequence. Although the speed of the robot in Vseq 4 is higher than Vseq 3, the false split corner rates are lower here. It is due to the

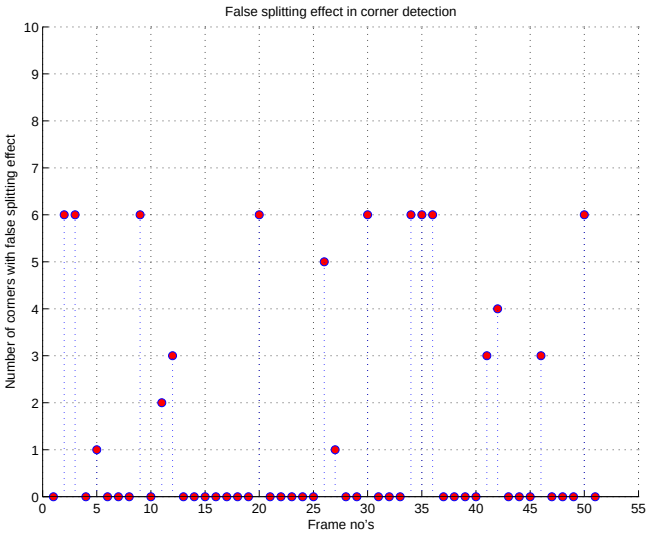


Figure 4.11: The false splitting effect in each frame of the third test video recorded while RHex was walking with 0.8 velocity coefficient at its fast walking mode.

high missed corner rate in Vseq 4. 14 frames are already lost completely and missed corners are seen 28 frames in total. These are two times higher than the missed corner rates in Vseq 3. This effect is mostly seen at the inner corners of pattern and a good way to parametrize the effect of motion blur especially when the frames are not degraded intensively. These results are demonstrated in the figure 4.11. Figure 4.11 shows the number of corners that are degraded by this effect in each frame.

In this video sequence, there is no false alarm which means a fake corner on the plate is not observed. This was achieved due to the carefully fixed threshold value of the algorithm. Besides, the texture of the image is not so complicated as to cause this kind of artificial features. .

As the last sequence, Vseq 5 is the fourth and the fastest motion video sequence that we used in our experiments. It was recorded while RHex was in its walking mode with 1.0 velocity coefficient. In order to see the performance of Harris Corner detector, we have analyzed missed corner rate, error rate of the detected corners, possible false alarms and false splitting effect on this video.

Missed corners appeared in only 46 frames among 51 frames and 30 frames among this 46 are completely lost which means that none of 18 corners can be detected by algorithm. Except

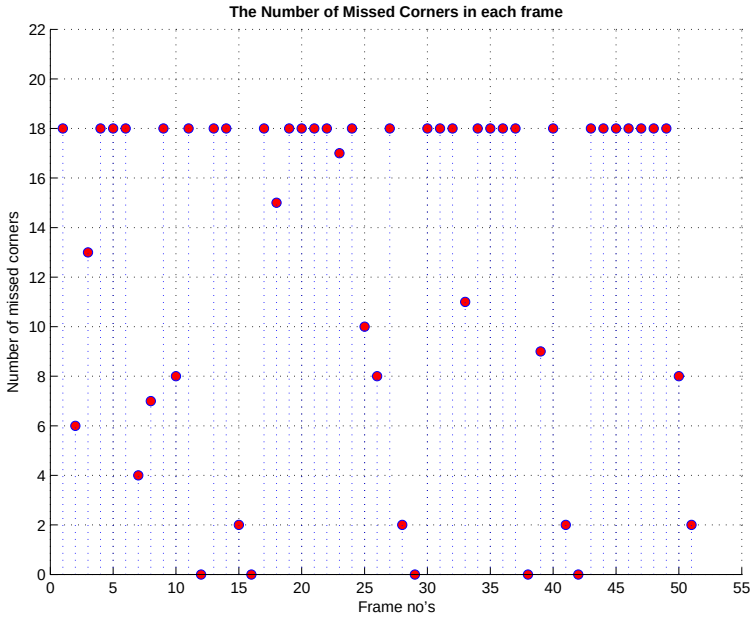


Figure 4.12: The number of missed corners in each frame of the fourth test video recorded while RHex was walking with 1.0 velocity coefficient at its fastest walking mode.

Table 4.6: The average rates per corner for Vseq 5

Average Missed Corner rate	14.4348
Average Strabismus rate	3.6250
Average Cross-eyed corner rate	14.4348

for these 30 frames, the average missed rate is 7.7500 in the other distorted frame. These results are demonstrated in the Figure 4.12. It shows that these video sequence is distorted completely.

The false splitting effect which means that single feature point response can not be achieved was seen in 15 frames among 51 frames in this video sequence. Although the speed of the robot in Vseq 5 is higher than Vseq 4 and Vseq 3, the strabismus rates are lower here. It is again due to the high missed corner rate in Vseq 5. 30 frames are lost completely and missed corners are seen in 46 frames in total which means that all sequence is almost lost in the sense of feature detection. These are two and three times higher than the missed corner rates in Vseq 4 and Vseq 3 respectively. These results are demonstrated in the figure 4.13. Figure 4.13 shows the number of corners that are degraded by this effect in each frame.

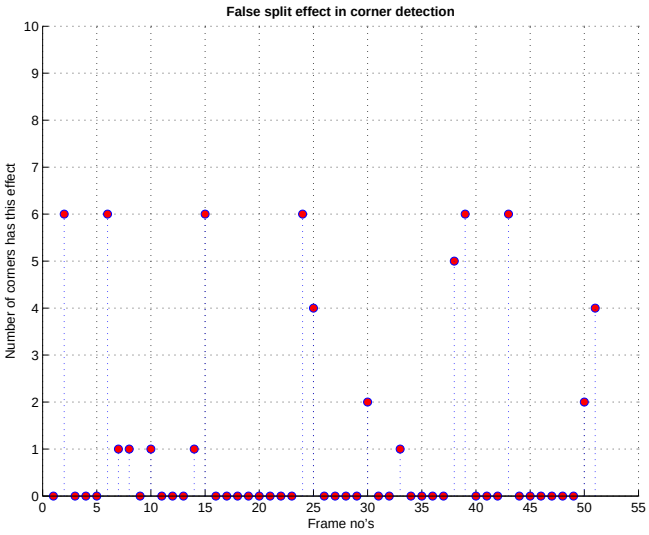


Figure 4.13: The false splitting effect in each frame of the fourth test video recorded while RHex was walking with 1.0 velocity coefficient at its fastest walking mode.

This effect is mostly seen at the inner corners of pattern. It gives good results while the intensity of blur is not so high. However, it is a good way to parametrize the effect of motion blur even if it is combined with missed corner rate. After threshold value of the algorithm is set in the first video, we haven't faced false alarms which means that there are no fake corners on irrelevant parts of the checkerboard plate during experiments. As in previous cases, this was achieved due to the carefully fixed threshold value of the algorithm and on the condition that the texture of the image is not so complicated as to cause this kind of artificial features.

4.4 Experiments with Canny Edge Detector

The first video which was recorded while robot was standing without moving is used as a reference data. The parameters of algorithm is set and tested according to this first video. And then, the performance of canny edge detector algorithm is investigated on other 4 test video sequences respectively.

51 frames and 24 edges in each frame for all 5 video sequences were taken into account for tests. The performance of canny edge detector were evaluated in two parts. Firstly as a blur metric and secondly as it was done for Harris corner detector.

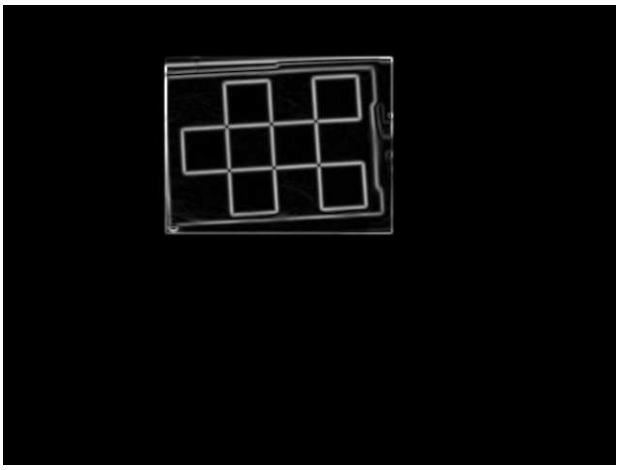
Edges were separated into two groups: the edges lying along the x-axis and the edges lying along the y-axis. The first part of the Canny algorithm contains calculating image derivatives which is called gradient image. As the first step of our performance evaluation, the gradient image was used with aim to observe the smoothing effect of blur on edges along x and y axes. Therefore, the thickness of edge locations were used as a motion blur metric of frames. As the second part of our performance evaluation, we have analyzed missed edge rate, possible false alarms and false splitting effect on frames. The first motionless video is used to calculate reference thickness of edges and to set the algorithm parameters.

The first video is motionless so it is used as reference video. The parameters of algorithm such as σ, T_H and T_L were set on this video sequence. Image derivatives were taken and the thickness of edges were calculated. One example frame and its derivative is shown in the Figure 4.14(b) and 4.14(b)

Vseq 2 is the first and the slowest motion video sequence that we used in our experiments. It was recorded while RHex was in its walking mode with 0.1 velocity coefficient. In order to observe the smoothing effect of blur on edges and its effects on the performance of Canny Corner detector, we have analyzed thickness of edges missed edge rate, possible false alarms and false splitting effect on this video.

As the first part of our experiment, the calculated edge thicknesses along x and y axes are shown in the Figure 4.15. The green line shows the reference thickness calculated on the motionless video sequence Vseq 1 and the red and green plots show average thickness of the edges along x axis and y axis respectively. It is observed that the blur along x axis is more

(a) One example frame of Vseq 1



(b) Derivative of this frame

Figure 4.14: One example frame and its derivative

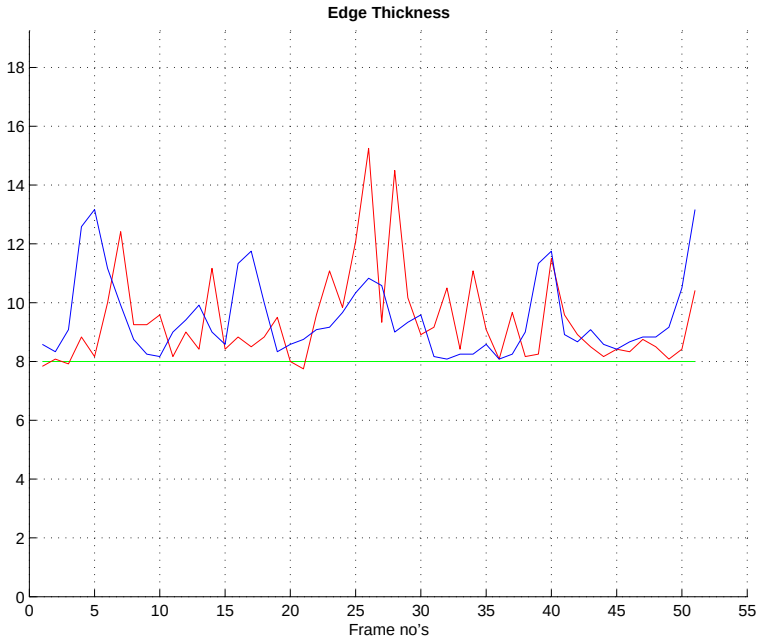


Figure 4.15: The average of edge thickness. Average thickness of edges along x axis is shown in red and average thickness of edges along y axis is shown in blue. The green shows the reference.

intensive than the blur along y axis for this video sequence Vseq 2.

Missed edges appeared in 2 frames among 51 frames and its average rate per frame is 3 in the 2 distorted frames. These results are demonstrated in the Figure 4.16. It shows that blur is not intensive in this sequence.

False edges due to noise in motion blurred frames appeared in only 6 frames among 51 frames and its average rate per frame is 1.33. These results are demonstrated in the Figure 4.17.

The false splitting effect which means that single edge response can not be achieved was seen in only 3 frames among 51 frames in this video sequence. It is an expected result because robot's velocity was really slow. Only at 5th and 51st frames, all the edges along x axis entirely degraded by this effect. These results are demonstrated in the Figure 4.18.

Vseq 3 is the second motion video sequence that we used in our experiments. It was recorded while SensoRHex was in its walking mode with 0.4 velocity coefficient. In order to observe the smoothing effect of blur on edges and its effects on the performance of Canny Corner detector, we have analyzed thickness of edges missed edge rate, possible false alarms and

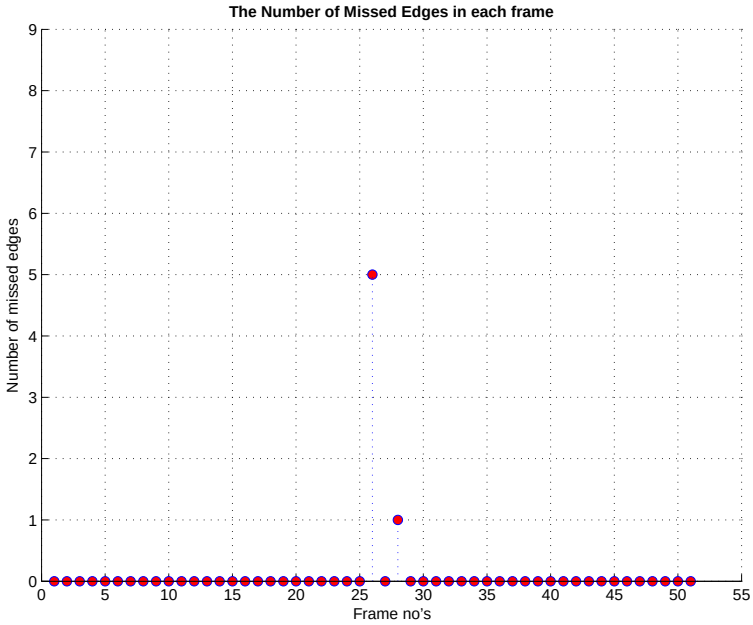


Figure 4.16: The number of missed edges in each frame of the second test video recorded while RHex was walking with 0.1 velocity coefficient at its slowest walking mode.

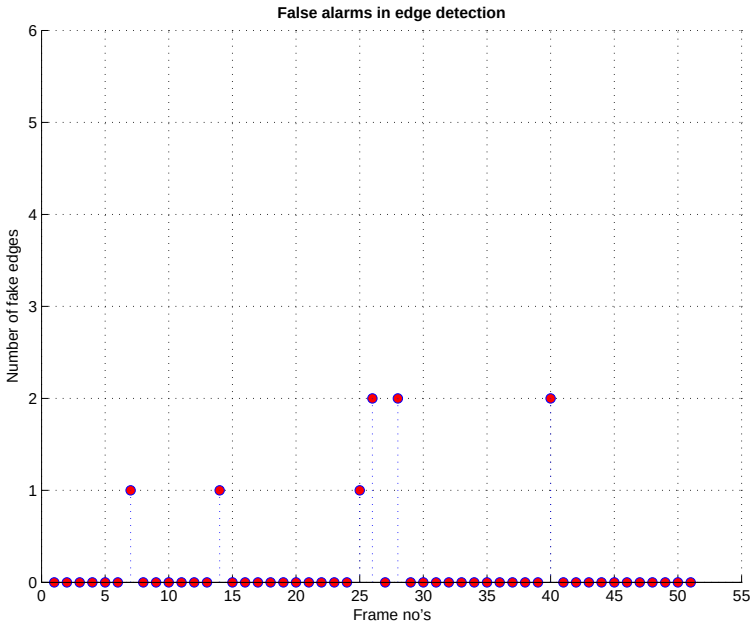


Figure 4.17: The number of False edges in each frame of the second test video recorded while SensoRHex was walking with 0.1 velocity coefficient at its slowest walking mode.

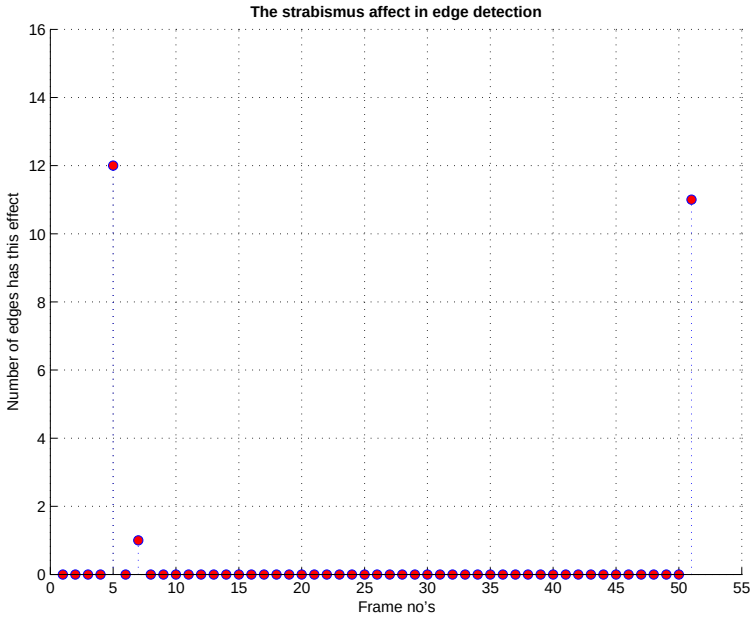


Figure 4.18: The number of edges on which false splitting effect is seen in each frame of the second test video recorded while SensoRHex was walking with 0.1 velocity coefficient at its slowest walking mode.

false splitting effect on this video.

As the first part of our experiment, the calculated edge thicknesses along x and y axes are shown in the Figure 4.19. The green line shows the reference thickness calculated on the motionless video sequence Vseq 2 and the red and green plots show average thickness of the edges along x axis and y axis respectively. It is observed that this video sequence contains moderate blur on both axis.

Missed edges appeared in 2 frames among 51 frames and its average rate per frame is 1.5 in the 2 distorted frames. These results are demonstrated in the Figure 4.20. It shows that there is not so intensive blur that may cause canny to fail in this sequence.

False edges due to noise in motion blurred frames appeared in only 10 frames among 51 frames and its average rate per frame is 1.9. These results are demonstrated in the Figure 4.20. We should consider on the missed edge rate and false rate edge together. Missed edge rate does not increase so fast because threshold values set more reluctantly but it causes false alarms.

The false splitting effect which means that single edge response can not be achieved were

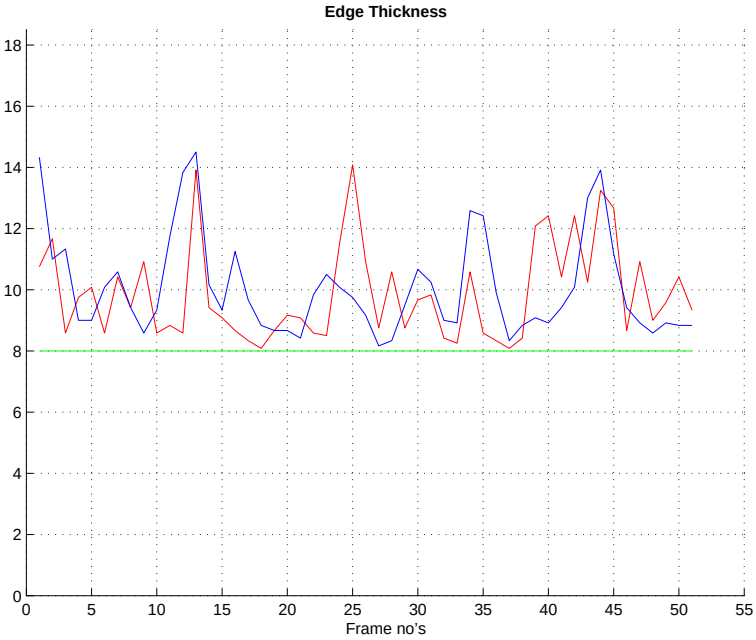


Figure 4.19: The average of edge thickness. Red line shows average thickness of edges along x axis and blue line shows average thickness of edges a long y axis. The green line shows the reference.

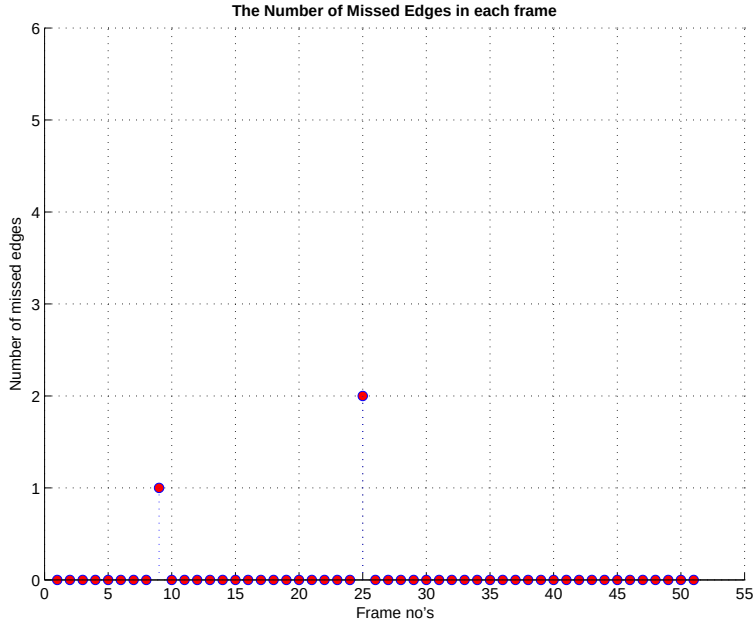


Figure 4.20: The number of missed edges in each frame of the second test video recorded while RHex was walking with 0.1 velocity coefficient.

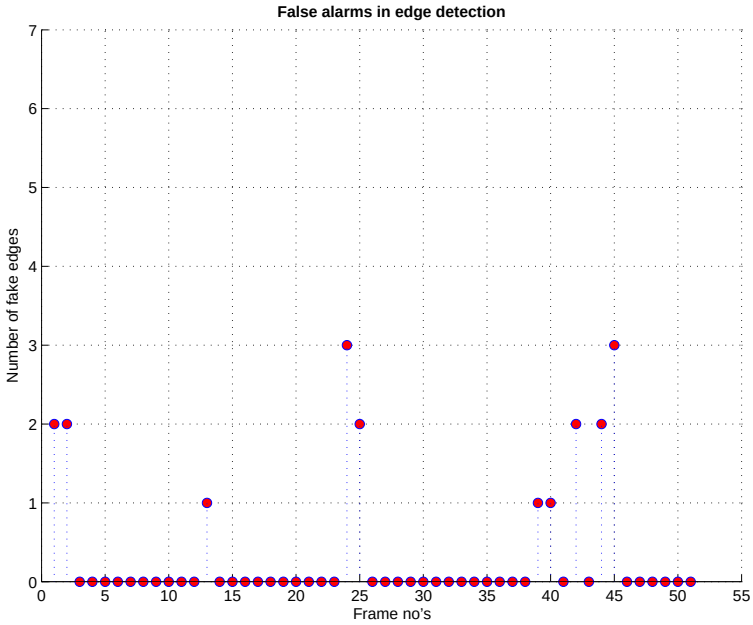


Figure 4.21: The number of False edges in each frame of the second test video recorded while RHex was walking with 0.4 velocity coefficient.

seen in only 7 frames among 51 frames in this video sequence. Its average is 5.71 an expected result because robot's velocity was really low. Almost all the edges along x axis degraded by this effect only at two frames. These results are demonstrated in the Figure 4.22.

Vseq 4 is the third motion video sequence that we used in our experiments. It was recorded while SensoRHex was in its walking mode with 0.8 velocity coefficient. In order to observe the smoothing effect of blur on edges and its effects on the performance of Canny Corner detector, we have analyzed thickness of edges missed edge rate, possible false alarms and false splitting effect on this video.

As the first part of our experiment, the calculated edge thicknesses along x and y axes are shown in the Figure 4.23. The green line shows the reference thickness calculated on the motionless video sequence Vseq 2 and the red and green plots show average thickness of the edges along x axis and y axis respectively. It is observed that this video sequence contains intensive blur on both axis. Especially the blur between 20 and 35th frames degrades the edges x-axis intensively.

Missed edges appeared in 15 frames among 51 frames and its average rate per frame is 4 in the 15 distorted frames. Moreover there are two frames in which half of the features are lost.

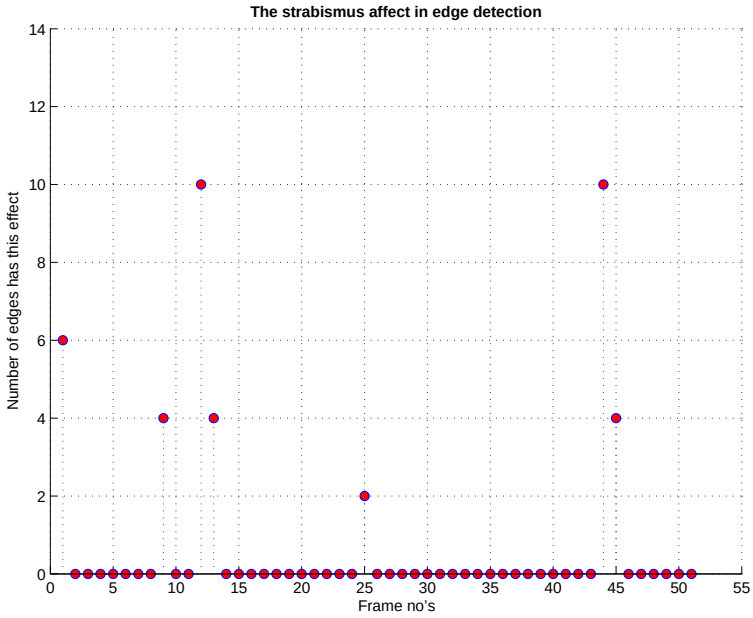


Figure 4.22: The number of edges on which false splitting effect is seen in each frame of the second test video recorded while SensoRHex was walking with 0.4 velocity coefficient at its walking mode.

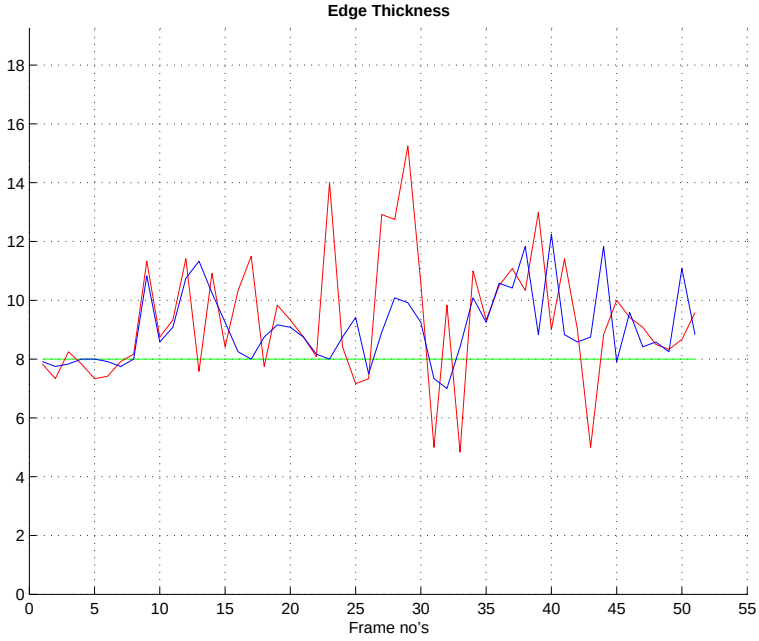


Figure 4.23: The average of edge thickness. Red shows average thickness of edges along x axis and blue shows average thickness of edges a long x axis. The green shows the reference.

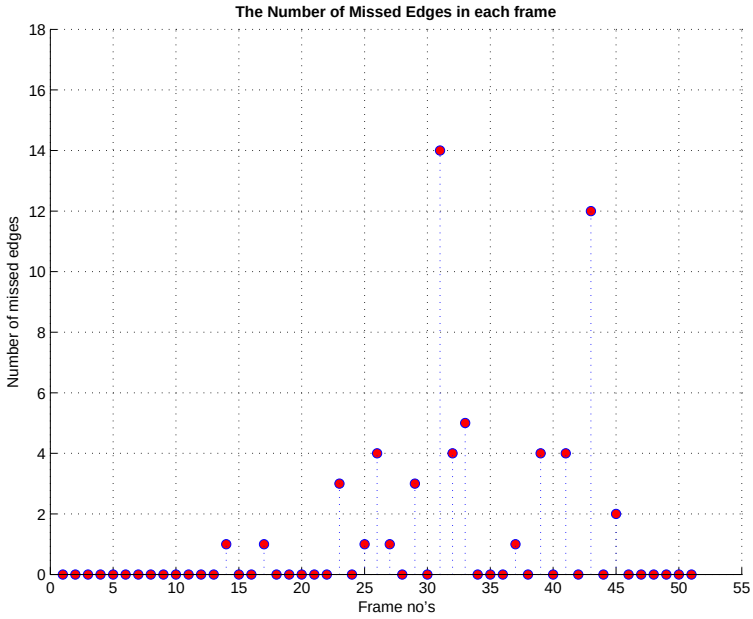


Figure 4.24: The number of missed edges in each frame of the second test video recorded while RHex was walking with 0.8 velocity coefficient.

These results are demonstrated in the Figure 4.24. It shows that there is intensive blur that may cause canny fails in this sequence.

False edges due to noise in motion blurred frames appeared in 25 frames among 51 frames which means half of the sequence and its average rate per frame is 2. These results are demonstrated in the Figure 4.25. That shows that this degradation is very common in video sequence.

The false splitting effect which means that single edge response can not be achieved was seen in only 19 frames among 51 frames in this video sequence. Its average is 3.47. Almost half of the sequences degraded by this and it is more intensive at two frames. These results are demonstrated in the Figure 4.26.

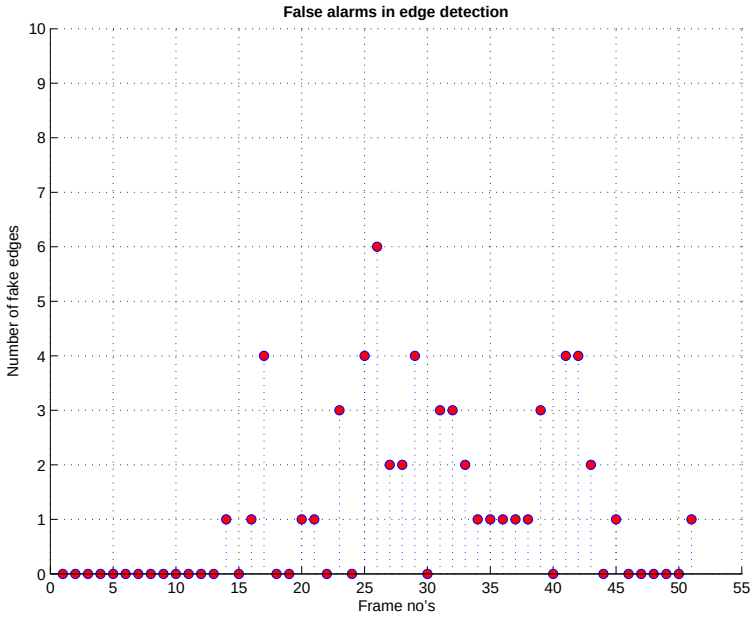


Figure 4.25: The number of False edges in each frame of the third test video recorded while RHex was walking with 0.8 velocity coefficient.

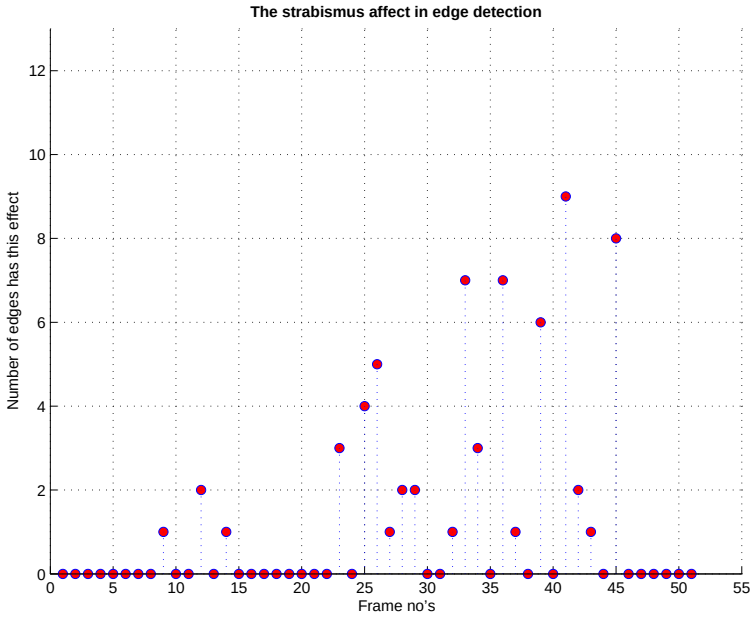


Figure 4.26: The number of edges on which false splitting effect is seen in each frame of the second test video recorded while SensoRHex was walking with 0.8 velocity coefficient at its fast walking mode.

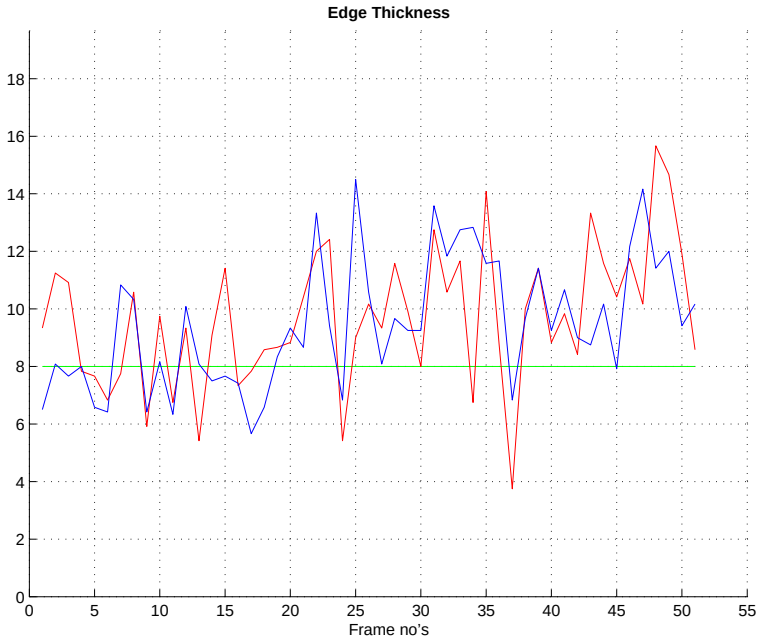


Figure 4.27: The average of edge thickness for the test video which has the highest velocity. Red line shows average thickness of edges a long x axis and blue line shows average thickness of edges a long y axis. Green line is the reference thickness if there is no motion

Vseq 5 is the last and the fastest motion video sequence that we used in our experiments. It was recorded while SesnoRHex was in its walking mode with 1.0 velocity coefficient. In order to observe the smoothing effect of blur on edges and its effects on the performance of Canny Corner detector, we have analyzed thickness of edges missed edge rate, possible false alarms and false splitting effect on this video.

İn the first part of our experiment, the calculated edge thicknesses along x and y axes, as shown in the Figure 4.27. The green line shows the reference thickness calculated on the motionless video sequence Vseq 5 and the red and green plots show average thickness of the edges along x axis and y axis respectively. It is observed that this video sequence contains intensive blur on both axis. Both of the axises are distorted by motion blur intensively.

Missed edges appeared in 36 frames among 51 frames and its average rate per frame is 6.97 in the 36 distorted frames. Moreover, there are 6 frames in which all edges are nearly lost. These results are demonstrated in the Figure 4.28. It shows that this sequence is completely distorted.

False edges due to noise in motion blurred frames appeared in only 41 frames among 51

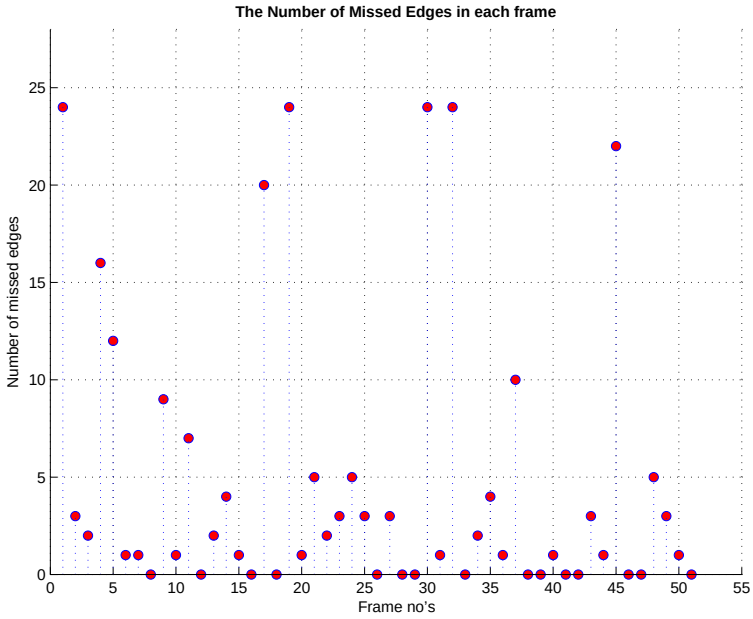


Figure 4.28: The number of missed edges in each frame of the second test video recorded while RHex was walking with 1.0 velocity coefficient.

frames and its average rate per frame is 2.82. These results are demonstrated in the Figure 4.29. These findings show that Canny edge detector can not work properly at all under this high amount of blur.

The false splitting effect which means that single edge response can not be achieved were seen in only 30 frames among 51 frames in this video sequence. Its average is 34.26. Moreover the average of false split edges is 9 and almost half of the sequences degraded by this although missed edges rates are so high too. These results are demonstrated in the Figure 4.30 which shows the number of edges that are degraded by this effect in each frame.

These figures show that this sequence recorded at the highest speed of walking mode of our experimental robotic platform SensoRHex is completely lost in the sense of edge detection.

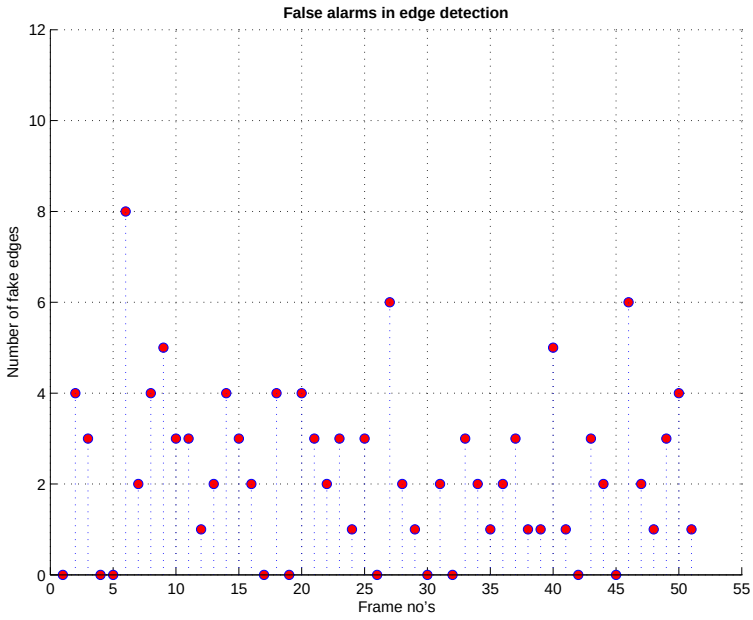


Figure 4.29: The number of False edges in each frame of the second test video recorded while RHex was walking with 1.0 velocity coefficient.

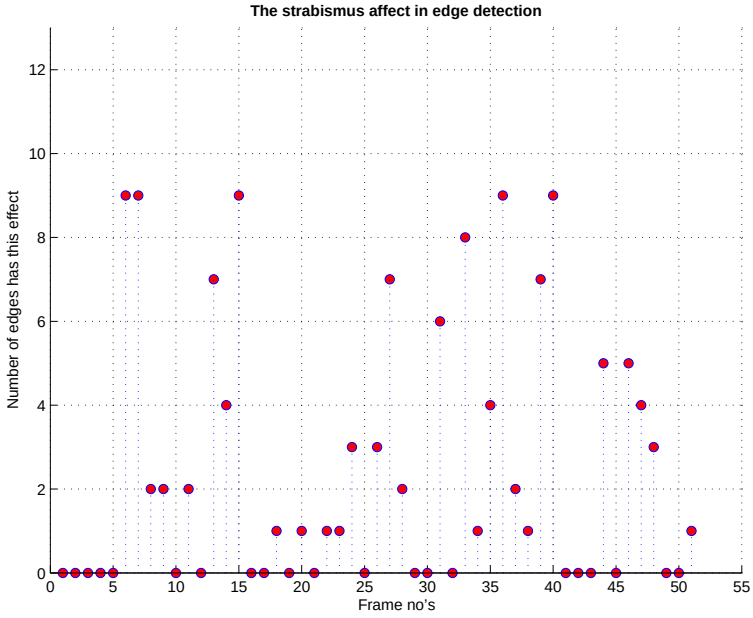


Figure 4.30: The number of edges on which false splitting effect is seen in each frame of the second test video recorded while SensoRHex was walking with 1.0 velocity coefficient at its fastest walking mode.

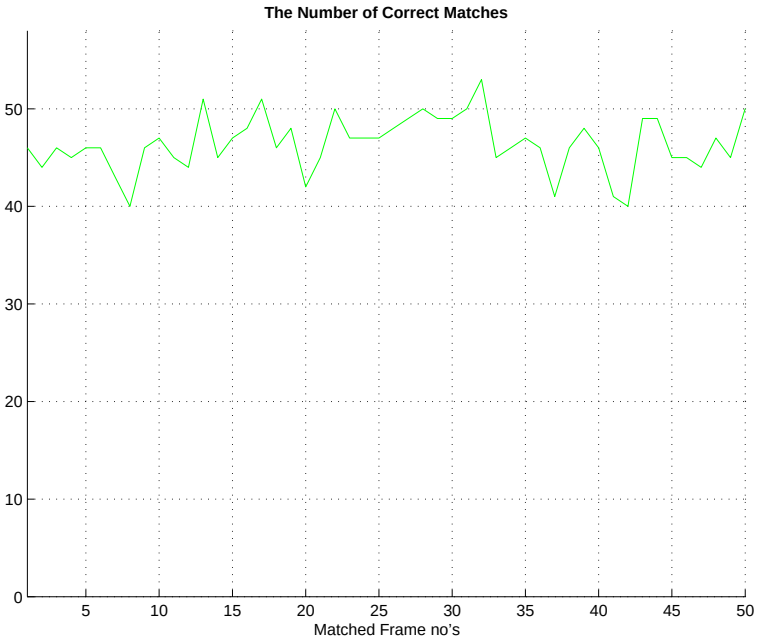


Figure 4.31: The number of SIFT matches for each frame without motion blur.

4.5 Experiments with SIFT

In this section, SIFT performance analyze in the context of matching is performed on the motion blurred frames. Although there are some works [43] which consider the effect of focus blur, there is not specific work which consider the effect of motion blur.

As we use for other feature detectors in the sections in Section 4.3 and Section 4.4, we use the same real test video sequences with four test locomotion velocities for sift performance analyze. The video sequence which is recorded without motion is used to show how algorithm works with sharp frames. The number of correct matches for each frame of test video without motion is demonstrated in the Figure 4.31.

The rate of false matches for each frame due to the noise is also calculated as a performance measure for SIFT algorithm. Then it is normalized by dividing it with the number of matches in this frame. The percentage of normalized false matches' rate is shown in the Figure 4.32. As it is expected, the percentage of miss matches is really so low on the sharp frames.

The first and the slowest motion video is recorded with the 0.1 velocity coefficient of our experimental robotic platform and in its walking mode. The number of correct matches for

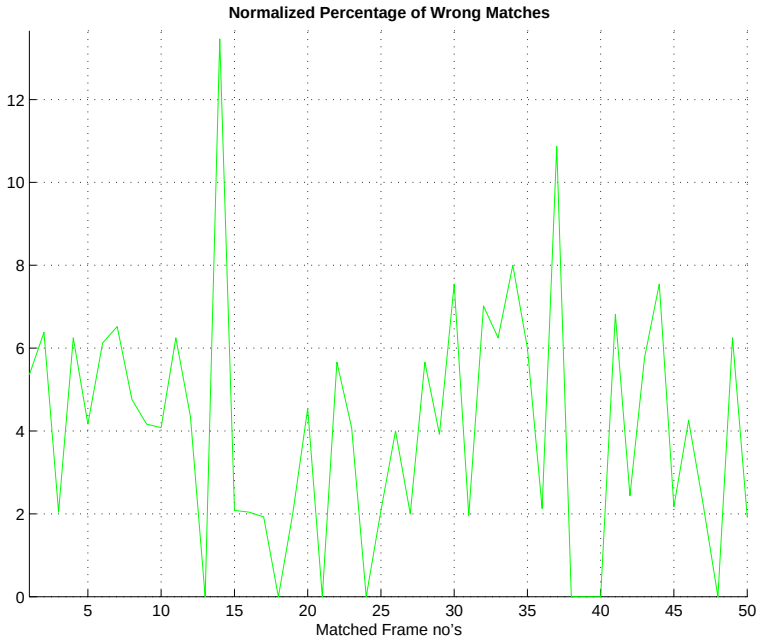


Figure 4.32: The percentage of SIFT normalized false matches for each frame without motion blur.

each frame of test video without motion is demonstrated in the Figure 4.33. As we can see from the figure 4.33, there is a significant difference with the values obtained with sharp frames although the velocity is low and degradation of motion blur is only visible in some individual frames.

The rate of false matches for each frame due to motion blur is also calculated and there is a sudden and unexpected increase in the percentage of the false matches if the low test velocity is considered. The percentage of normalized false matches' rate is shown in the Figure 4.34. Moreover the percentage of false match rate is above 50%. It is because this match contains two intensively blurred consecutive frames as it is shown in figure 4.35. Although the velocity of the robot is low, the reason of this intensive blur is assumed that the camera catch the exact moment when the front leg hit the ground. Due to the flexible c-shape legs, these moments when the leg thouch the ground or leaving the ground creates extra shocks and disturbance on the robot's platform.

The second motion video with 0.4 velocity coefficient of our experimental robotic platform is used for the same analysis. The number of correct matches continues to decrease as the velocity of the robotic platform increases. The difference of correct matches rate between the

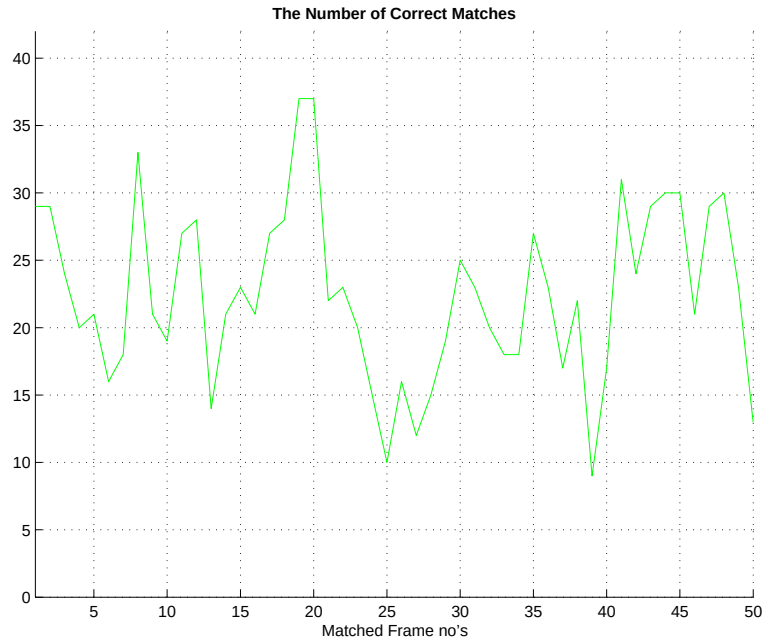


Figure 4.33: The number of SIFT matches for each frame of the test video with 0.1 test locomotion velocity.

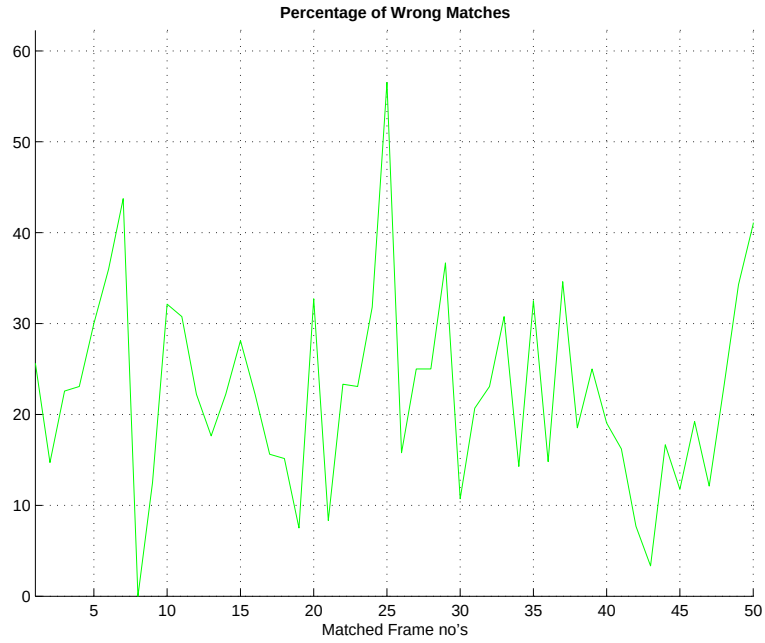


Figure 4.34: The percentage of SIFT false matches for each frame of the test video with 0.1 test locomotion velocity.

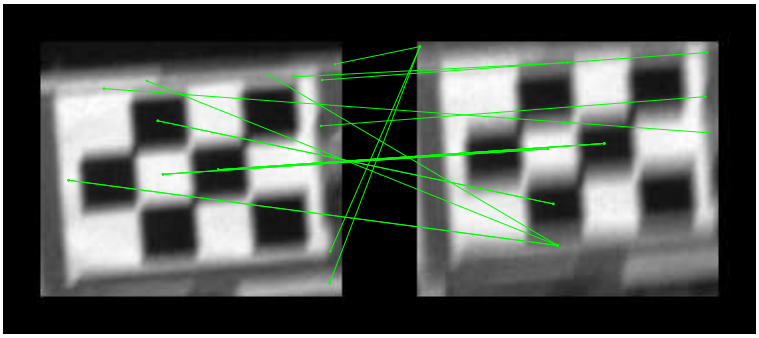


Figure 4.35: The 25th match in the video sequence with 0.1 test locomotion velocity. The percentage of SIFT false matches for these frames is 56.52 which is unexpectedly higher than the average of sequence.

frames with 0.1 velocity and frames with 0.4 velocity is not as big as the difference between frames without motion and frames with 0.1 velocity but still there is a meaningful decrease. The results are shown in the Figure 4.36.

If the percentage of false sift matches is considered, it can be seen in the Figure 4.37. The first half of the video sequence is degraded by motion blur more than the second half. Although motion blur effects many frames, there are still well-conditioned frames in the sense of feature detection between heavily blurred consecutive frames. This gives chance to develop interpolation/smoothing approaches where the feature computation for the corrupted frame is corrected by means of considering surrounding frames instead of applying more complex deblurring algorithms.

Vseq4 is the video sequence recorded while our experimental robotic platform is in the fast walking mode. Figure 4.38 demonstrates the correct match rate and figure 4.39 demonstrates the percentage of missed matches. As it is seen from the figures, there are even four completely missed frames in the sense of SIFT matches. Especially after the first half of the sequence, we can say that all frames are heavily corrupted by motion blur. Therefore, it is not so feasible to consider interpolation methods especially for the second half of the sequence.

Figure 4.40 is given as an example of one type of completely missed frames in this sequence. In this case, both frames are heavily blurred and it is impossible to make any match. The other type of completely missed frames is one frame heavily blurred and following frame is relatively better but sift algorithm fails as in figure 4.41.

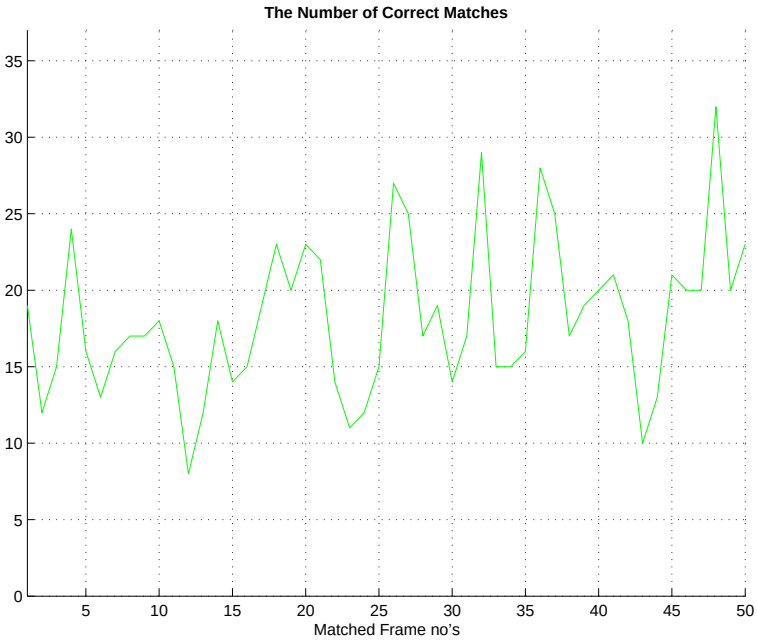


Figure 4.36: The number of SIFT matches for each frame of the test video with 0.4 test locomotion velocity.

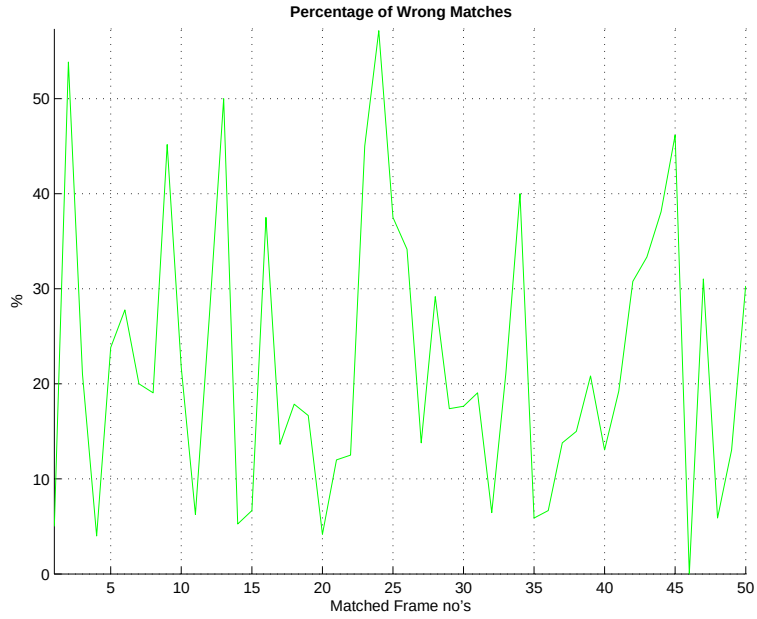


Figure 4.37: The percentage of SIFT false matches for each frame of the test video with 0.4 test locomotion velocity.

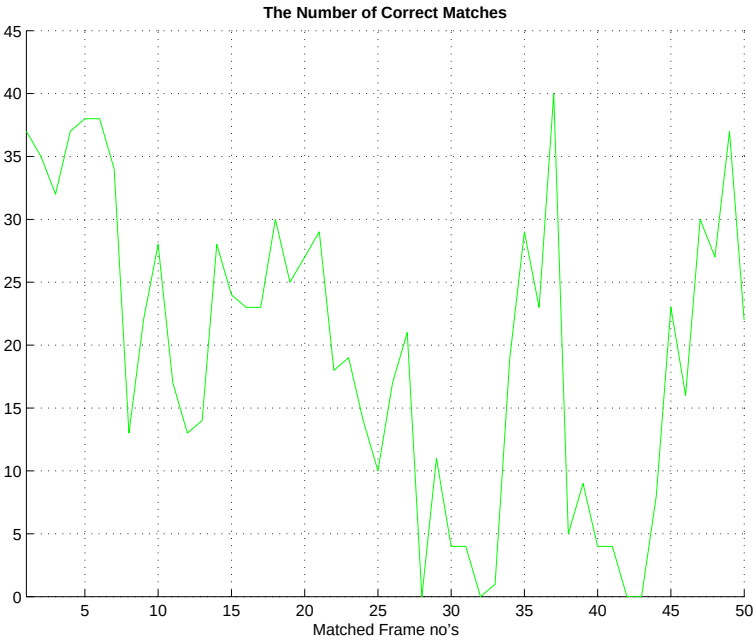


Figure 4.38: The number of SIFT matches for each frame of the test video with 0.8 test locomotion velocity.

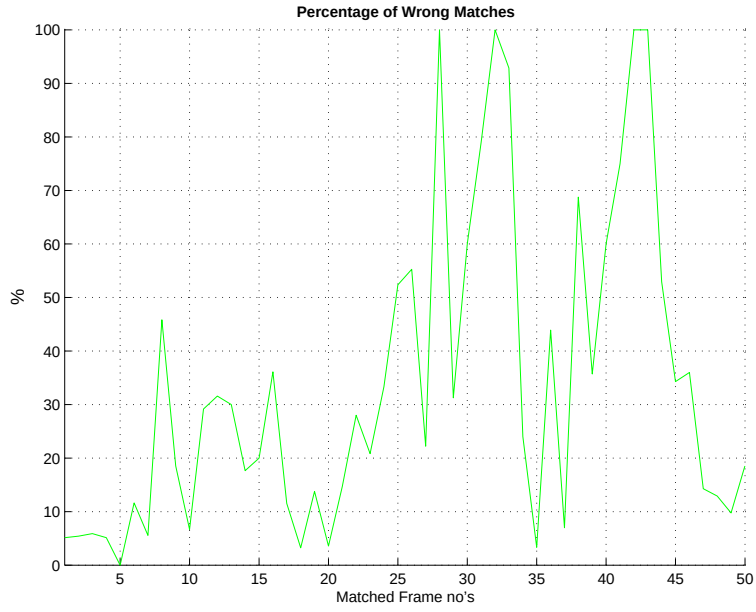


Figure 4.39: The percentage of SIFT false matches for each frame of the test video with 0.8 test locomotion velocity.

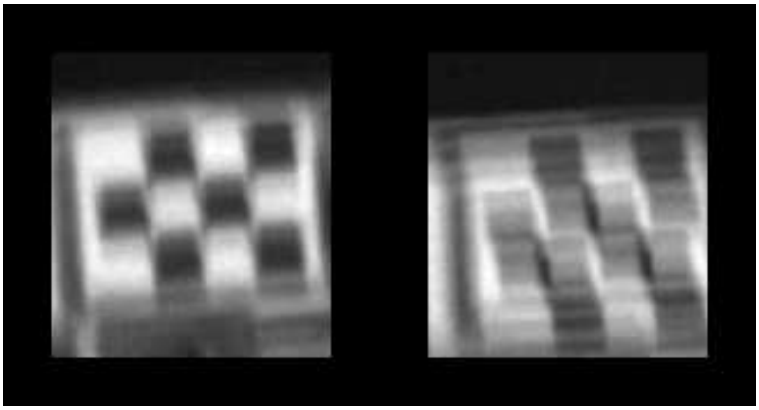


Figure 4.40: Example to the first type of completely missed match in this sequence with 0.8 velocity. Both of the frames are heavily blurred.

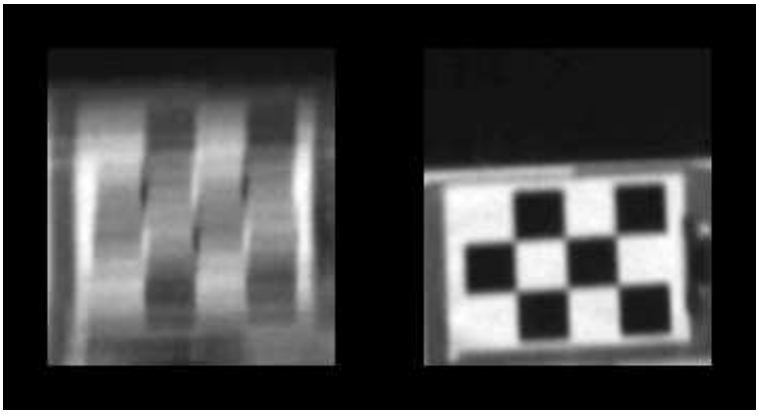


Figure 4.41: Example to the second type of completely missed match in this sequence with 0.8 velocity. Both of the frames are heavily blurred.

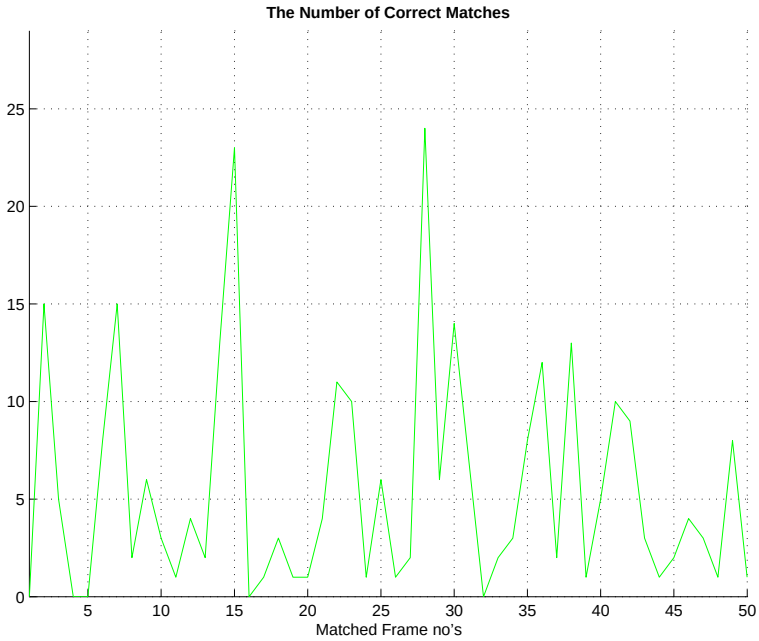


Figure 4.42: The number of SIFT matches for each frame of the test video with 1.0 test locomotion velocity.

Vseq5 which is recorded while our experimental robotic platform is walking at its fastest mode. Figure 4.42 and figure 4.43 show that this sequence is all heavily blurred and SIFT can not give relevant and robust result except of 10 frames among 51 frames.

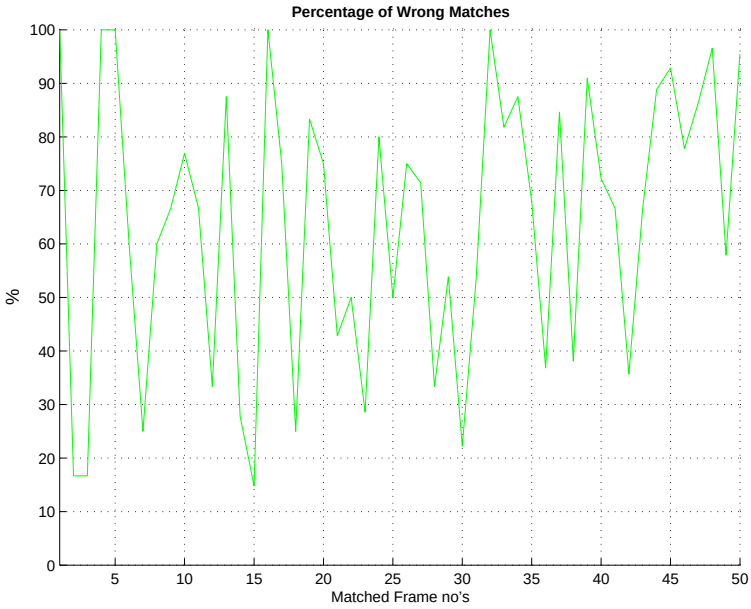


Figure 4.43: The percentage of SIFT false matches for each frame of the test video with 1.0 test locomotion velocity.

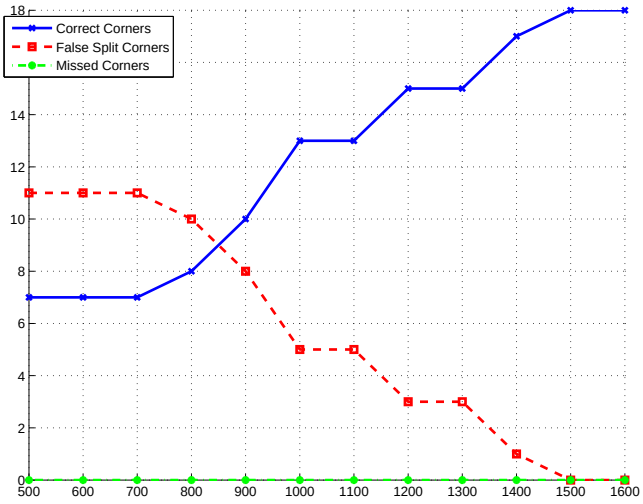


Figure 4.44: Sensitivity analysis of Harris corner detector based on threshold for the most blurred frame of the first test video recorded while our experimental robotic platform RHex was walking with 0.1 velocity coefficient at its slowest walking mode.

4.6 Input Parameters Analysis of Feature Detectors

We tried to fix the parameters manually based on obtaining minimum number of fake detections in our experimental tests. Limited number of frames used in the tests and limited number of parameters in selected algorithms allowed us to have done this without using extra optimization methods.

In this section, one form of sensitivity analysis of feature detectors is given for each feature detector algorithm. The first algorithm used in our experimental tests was Harris corner detector. Variance of Gaussian as smoothing function and a threshold value were the input parameters chosen by the user. Due to the fact that we were dealing with motion blurred frames, we chose the variance parameter of smoothing function minimum. And then the changes in threshold parameter left as an effect on the performance of the algorithm.

The performance of Harris corner detector is given in a determined threshold interval as $[500, 1600]$. To be able to show this, one among the most blurred frames and one among the most sharpest frames were chosen in each video sequence. Figures 4.44 and 4.45 show the performance of algorithm on the first test video while threshold values are changing in the given interval.

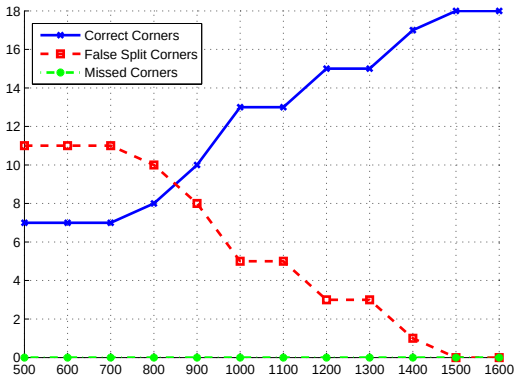


Figure 4.45: Sensitivity analysis of Harris corner detector based on threshold for one of the sharpest frame of the first test video recorded while our experimental robotic platform RHex was walking with 0.1 velocity coefficient at its slowest walking mode.

Figures 4.46 and 4.47 show the performance of algorithm on the first test video while threshold values are changing in the given interval.

Figure 4.48 shows the performance of algorithm on the first test video while threshold values are changing in the given interval. For this sequence, the highly blurred frames are completely lost on the sense of corner detection. Even at small threshold values, algorithm can not find more than 2-3 correct corners among 18 corners. Therefore, only one moderately blurred frame is used for this analysis.

Although the number of the missed corners are less at small threshold values, this causes high number of false split corners at the sharp and degraded by moderate blur frames. Therefore, we prefer high threshold values during our experimental tests with Harris corner detector.

This analysis is also done for Canny Edge Detector. Again, variance of smoothing function is set to be the minimum value because we are working with motion blurred frames. Different than Harris corner detector, canny uses two threshold values which are given as high and low. This test is done between the intervals such as for the low threshold $[0.01 : 0.1]$ and for the high threshold value $[0.1 : 1.0]$.

The results for the first video are given in the figures 4.49 and 4.50.

Figures 4.51 and 4.52 show how the performance of the algorithm changes on the second test video sequence.

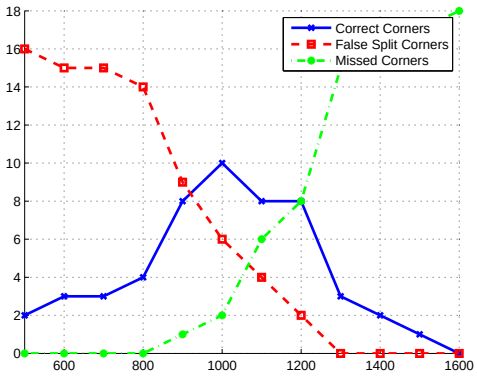


Figure 4.46: Sensitivity analysis of Harris corner detector based on threshold for the most blurred frame of the test video recorded while our experimental robotic platform RHex was walking with 0.4 velocity coefficient at its slowest walking mode.

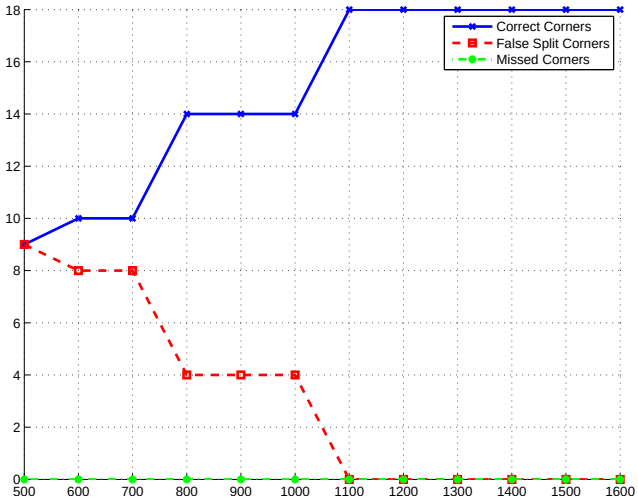


Figure 4.47: Sensitivity analysis of Harris corner detector based on threshold for one of the sharpest frame of the test video recorded while our experimental robotic platform RHex was walking with 0.1 velocity coefficient at its slowest walking mode.

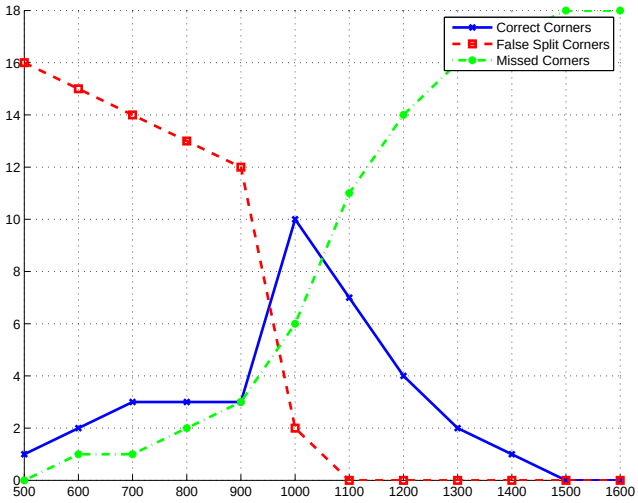


Figure 4.48: Sensitivity analysis of Harris corner detector based on threshold for one of the blurred frame of the test video recorded while our experimental robotic platform RHex was walking with 0.1 velocity coefficient at its slowest walking mode.

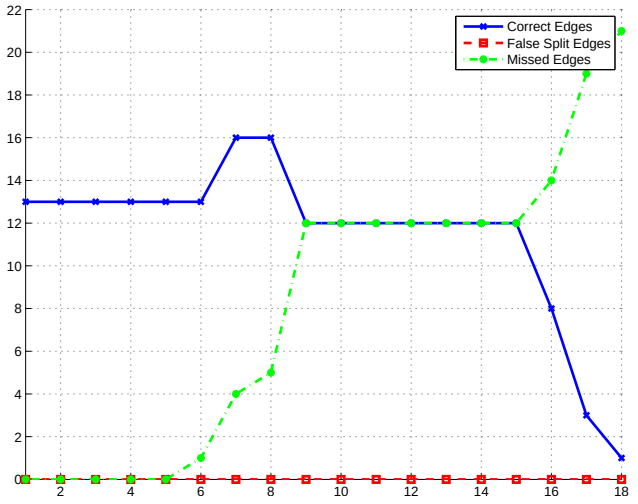


Figure 4.49: Sensitivity analysis of Canny edge detector based on threshold for the most blurred frame of the first test video recorded while our experimental robotic platform RHex was walking with 0.1 velocity coefficient at its slowest walking mode.

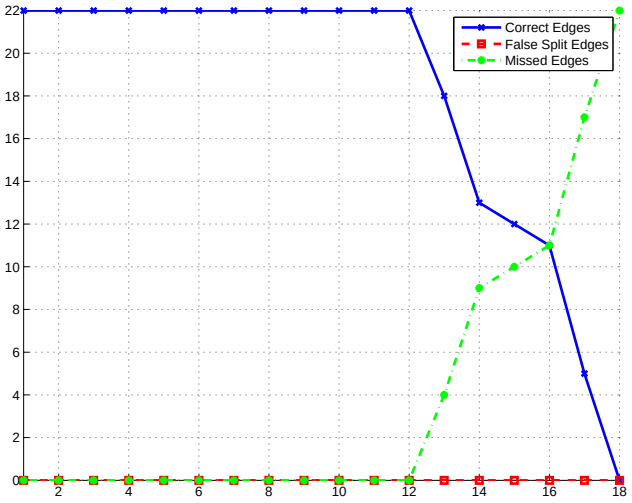


Figure 4.50: Sensitivity analysis of Canny edge detector based on threshold for one of the sharpest frame of the first test video recorded while our experimental robotic platform RHex was walking with 0.1 velocity coefficient at its slowest walking mode.

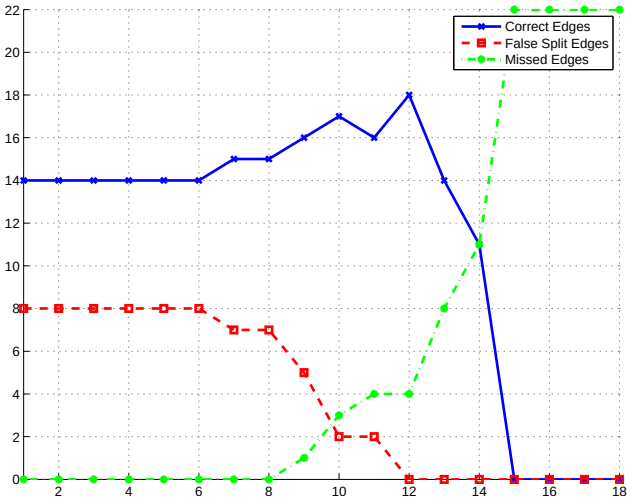


Figure 4.51: Sensitivity analysis of Canny edge detector based on threshold for the most blurred frame of the test video recorded while our experimental robotic platform RHex was walking with 0.4 velocity coefficient at its slowest walking mode.

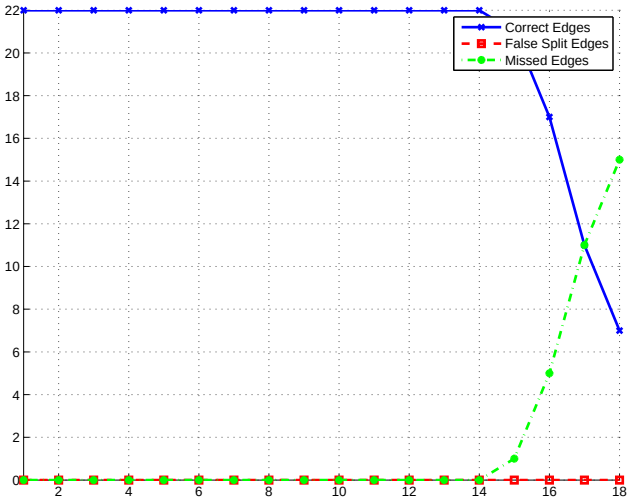


Figure 4.52: Sensitivity analysis of Canny edge detector based on threshold for one of the sharpest frame of the test video recorded while our experimental robotic platform RHex was walking with 0.1 velocity coefficient at its slowest walking mode.

For the other two videos which were recorded at higher velocities of our robotic experimental platform, the most blurred frames do not give us any information for this analysis. Giving small threshold values increases fake detection and does not show any improvement to find any correct edges. Therefore, only moderately blurred frames in these video sequences are used for this test. Figure 4.53 shows the results of one of the sharp frame which is so rare in the fastest sequence.

The figures show that finding the optimal threshold value value for our experiments are mostly obvious especially if the most important criteria is to obtain small number of fake features. Low threshold values increases the detected corner even in the some of highly blurred frames but it cases many fake corner in other moderately blurred or less blurred frames. In fact having no feature in the highly blurred frames and having less correct features but without fakes is more valuable for our evaluation and blur estimation idea. In our case, we prefer high threshold values with this reason. The code of Lowe is implemented with his given optimum parameters for SIFT therefore we haven't made any extra sensitivity analysis for this algorithm.

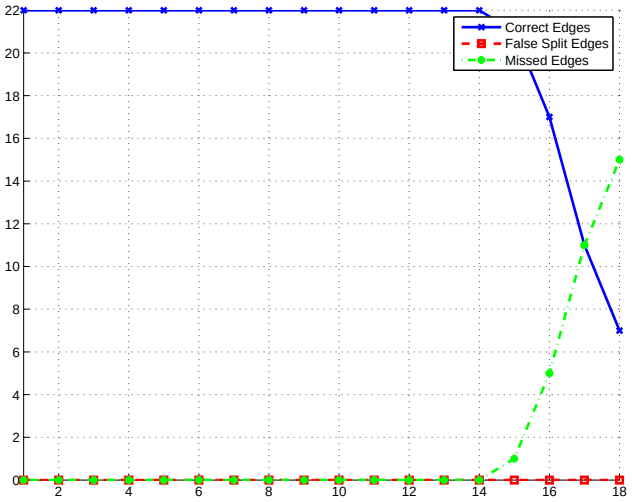


Figure 4.53: Sensitivity analysis of Canny edge detector based on threshold for one of the sharpest frame of the test video recorded while our experimental robotic platform RHex was walking with 1.0 velocity coefficient at its slowest walking mode.

4.7 Discussions on Experimental Results

Systematic experimental characterization of motion blur effect on well-known and popular feature detectors such as Harris corner detector, Canny edge detector and SIFT was performed. The severity of motion blur investigated in 4 videos by changing the speed parameters of our experimental robotic platform. To be able to classify the results, new degradation types were proposed on frames.

The results related with Harris Corner Detector were categorized under two groups. The results were obtained while experimental robotic platform were walking in slow speeds such as 0.14 m/sec and 0.16 m/sec and the results were obtained while experimental robotic platform were walking in fast walking speeds such as 0.25 m/sec and 0.40 m/sec.

In the first case, the main reason of the highly degraded image featured is due to vibrations and shocks. These vibrations caused by the leg locomotion of our experimental hexapod robot and the shocks occurred if the legs of robot touches the ground during exposure time of the camera. We analyzed only two seconds of the video due the limitations of our short running path and limited field of view of the camera and these kind of highly damaged frames were observed one or two times among this 2 second video part. The most valuable degradation type among the proposed one is the number of missed corners. After some amount of motion

blur, we could not observe false split corners because we almost lost the texture of the target plain. The number of false split corners gives us information if the frame is degraded by moderate motion blur. This effect mostly appeared in the inner corners of the checkerboard plate. In fact, it was firstly seen at inner corners and then at outer corners proportional to the amount of blur. The other reason of motion blur based on the speed of robot is so limited in these video sequences because of the slow walking of our experimental robotic platform.

In the second case, most of the frames are highly degraded by motion blur. Due to fact that our experimental platform walked in high speeds, the observed movements especially in roll and pitch angle is high. And when this movements combined with shocks explained above, this resulted in more common and more severe corner feature degradation in the frames. Missed corners and false split corners observed almost all the frames in these two video sequences. Therefore, the combination of the number of missed corners and false split corners gave us information about the amount of the motion blur. False alarms never observed on the results of Harris corner detector. This was mostly related with the texture of the target plain and the chosen threshold value of the algorithm.

The results related with Canny edge detector were also investigated as it was done for Harris corner detector. In addition to these, the first part of the Canny edge detector that gave the derivatives of the image along x and y axes used for estimation of the amount of motion blur. These estimation results were relevant with the results that were obtained from the number of missed edges and the number of false split edges. Especially, if edge thicknesses in a frame are high in both directions, then this frame had also high number of missed and false split edges.

The results related with SIFT were categorized with different performance measure. The matching rate between the frames in our 4 different video sequences was investigated and the performance measure defined on this criteria. The number of correct matches and the percentage of false matches were calculated for each frame. Although this algorithm claims that it is robust to the motion blur, there is a sudden change between motionless and motion video sequences. Especially if there is intense motion blur, it completely failed to match the features on the target plain.

CHAPTER 5

CONCLUSIONS

In this thesis, we performed systematic experimental characterization of motion blur effect on well-known and popular feature detectors. The aim was to analyze the effect of motion blur for the purposes of perception and to plan which methods would be necessary to achieve robustness against motion blur. Our aim was to develop a technique which is robust under the legged motion and the resulting visual disturbances. Therefore, experimental performance evaluation of Harris corner detector, Canny edge detector and SIFT in the existence of real motion blur was presented here.

We first presented a survey of existing models of motion blur and approaches to motion deblurring and reviewed recent literature on motion blur and deblurring in Chapter 2. We focused our attention on motion blur induced degradation of a number of popular feature detectors. In Chapter 3 we presented the necessary background for Harris corner detector, Canny edge detector and SIFT, and an overview of the effects of motion blur on these feature detectors. The results of the study supported that motion blur is clearly an undesirable phenomenon in computer vision not only because it degrades the quality of images but also causes other feature extraction procedures to degrade or fail.

The performance degradation of these feature detectors due to motion blur were categorized to analyze the effect of legged locomotion on feature performance for perception in Chapter 4. In our tests, we used 5 real video sequences which were captured by the vision system of a mobile legged robot platform. We have put a target pattern in the field of view of the robot camera and focused on the features of this pattern in our experiments. These analyses results were classified under proposed degradation types such as missed or detected features, false alarm, false split features, correct match rate and percentage of false matches. All these find-

ings were obtained as a first step towards the stabilization and restoration of video sequences captured by our experimental legged robotic platform and towards the development of motion blur robust vision system.

The performance of feature detectors were almost always degraded by motion blur. Moreover, the experiments showed that there are two types of motion blur effect on our video sequences which depend on the velocity of robot platform. The first effect is seen at low velocities such as 0.1 and 0.4. Motion blur in these video sequences effected only a small number of consecutive frames, or even a singular frame which were/was surrounded by relatively sharp frames in the sense of feature detection and image quality. According to us, these degraded frames which appear unexpectedly in the sequence, happened due to the shocks and stretching effect of c-shape legs when they touch the ground. The second effect was seen at high velocities such as 0.8 and 1.0. This time motion blur affected more than a small number of frames. These video sequences contained big number of corrupted frames, or almost all frames in the sequences were corrupted. This had two reasons: the shocks and stretching effect of c-shape legs increased while their turning periods increased and the relative motion between the scene and the camera increased while robot platform started moving faster.

Finally, we considered the performance of selected well-known feature detectors under motion blur effect due to legged locomotion. Our future work is planned on achieving motion blur robust feature detectors especially for the video sequences in which the first type of motion blur effect is seen. As for the second type of motion blur effect, we are planning to continue our studies on improving inverse filtering method based on accurate motion estimate which can be obtained by the help of inertial measurement unit.

5.1 Future Work

The thesis work is part of our long term objective of working with visual sensor data on legged robots where serious platform motion effect the quality of the captured video frames. Motion Blur is one of the most important degradations that have an effect on the performance of subsequent perception algorithms. As a follow up to the present work where we have attempted to carefully characterize the effects of motion blur on some important feature extraction algorithms, we need to continue our investigation in a number of directions where we will attempt

to compensate for motion blur.

Firstly, based on the observation that motion blur sometimes heavily effect only a small number of consecutive frames, or even a singular frame, we will attempt to develop interpolation/smoothing approaches where the feature computation for the corrupted frame is corrected by means of considering surrounding frames.

For cases where more than a small number of frames are corrupted or cases where the high quality frame images themselves are required for the robot user, we will consider motion blur correction techniques for individual frames or frame sequences as surveyed in the survey section of the thesis. These approaches are mostly based on inverse filtering techniques where an accurate estimation of the motion is very important.

In fact, there may be two fundamental different approaches to blur compensation depending on the application requirements. For the case images are only required for the subsequent features, the first approach may directly compensate/correct the features themselves rather than deblurring the source image. Feature interpolation just discussed belongs to this category. A second approach for application where the video frames are themselves required (e.g. operator surveillance), deblurring or inverse filtering of frames should be followed by better feature extraction.

Although the motion can often be estimated to some degree from the individual frame itself or the sequence of frames, we have plans to consider the use of an Inertial sensor such as an Inertial Measurement Unit (IMU) to estimate the corrupting motion. This is a sensor readily available on most robotic platforms and is able to measure accelerations along the three body axes and turn rates around them. Hence, the sensor is very powerful for accurately measuring motion leading to the motion blur. We hope an accurate estimation of the motion will give us a better chance of constructing successful inverse filters.

We are aware that motion blur is a non-invertible corruption when a single frame is considered. However, there are approaches in the literature that we have outlined in Chapter 2 that can consider multiple frames with certain properties in order to recover an individual frame with much higher success. We hope to integrate these approaches into our robotic application to achieve a much more robust vision sub-system in the presence of legged motion and the resulting visual disturbances.

REFERENCES

[1] E. H. Adelson and J. Y. A. Wang. Representing moving images with layers. *IEEE Transactions on Image Processing*, 3:625–638, 1993.

[2] S. R. R. N. S. K. Adrien Stern, Etan Fisher. Spie influence of severe vibrations on the visual perception of video sequences. In *Infrared Technology and Applications XXVI*, volume 4130, pages 727–733, 2001.

[3] A. Agrawal, Y. Xu, and R. Raskar. Invertible motion blur in video. *ACM Trans. Graph.*, 28(3):1–8, 2009.

[4] N. J. P. R. Bardsley J., Jefferies S. Blind iterative restoration of images with spatially-varying blur. In *Optics Express*, pages 1767–1782, 2006.

[5] B. Bascle, A. Blake, and A. Zisserman. Motion deblurring and super-resolution from an image sequence. In *Proceedings of the 4th European Conference on Computer Vision-Volume II*, pages II:571–582, 1996.

[6] M. Ben Ezra and S. Nayar. Motion deblurring using hybrid imaging. pages I: 657–664, 2003.

[7] M. Ben Ezra and S. Nayar. Motion-based motion deblurring. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):689–698, June 2004.

[8] P. Bhat, C. L. Zitnick, N. Snavely, A. Agarwala, M. Agrawala, B. Curless, M. Cohen, and S. B. Kang. Using photographs to enhance videos of a static scene. In J. Kautz and S. Pattanaik, editors, *Rendering Techniques 2007 (Proceedings Eurographics Symposium on Rendering)*, pages 327–338. Eurographics, jun 2007.

[9] G. Boracchi, A. Foi, V. Katkovnik, and K. Egiazarian. Deblurring noisy radial-blurred images: spatially-adaptive filtering approach. In *Electronic Imaging, Science and Technology, January 2008, San Jose, California, USA*, 2008.

[10] G. J. Brostow and I. Essa. Image-based motion blur for stop motion animation. In E. Fiume, editor, *ACM SIGGRAPH Proceedings of Annual Conference on Computer graphics and interactive techniques*, Computer Graphics Proceedings, Annual Conference Series, pages 561–566. ACM, ACM Press / ACM SIGGRAPH, 2001. Appears in ACM Transactions on Graphics (TOG).

[11] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698, 1986.

[12] C.-H. Chen, T. Chien, W.-C. Yang, and C.-Y. Wen. Restoration of linear motion and out-of-focus blurred images in surveillance systems. In *IEEE International Conference on Intelligence and Security Informatics*, pages 239–241, 2008.

[13] J. Chen, L. Yuan, C. Tang, and L. Quan. Robust dual motion deblurring. In *IEEE COMPUTER SOCIETY CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION*, pages 1–8, 2008.

[14] W.-G. Chen, N. Nandhakumar, and W. N. Martin. Image motion estimation from motion smear-a new computational model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:412–425, 1996.

[15] S. Cho, Y. Matsushita, and S. Lee. Removing non-uniform motion blur from images. *Computer Vision, IEEE International Conference on*, 0:1–8, 2007.

[16] P. Csillag and L. Boroczky. Estimation of accelerated motion for motion-compensated frame interpolation. In A. R. S. MJT, editor, *SPIE*, volume 2727 of *The Society of Photo-Optical Instrumentation Engineers (SPIE)*, pages 604–614, 1996.

[17] S. Dai and Y. Wu. Motion from blur. In *In Proc. Conf. Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[18] P. Favaro and S. Soatto. A variational approach to scene reconstruction and image segmentation from motion-blur cues. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1:631–637, 2004.

[19] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman. Removing camera shake from a single photograph. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, pages 787–794, New York, NY, USA, 2006. ACM.

[20] J. Fox. Range from translational motion blurring. In *IEEE Conferance on Computer Vision and Pattern Recognition*, pages 360–365, 1988.

[21] A. Glassner. An open and shut case. *IEEE Computer Graphics and Applications*, 19:82–92, 1999.

[22] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.

[23] S. Hammett. Motion blur and motion sharpening in the human visual system. *VISION RESEARCH*, 37:2505–2510, 1997.

[24] P. C. Hansen, J. G. Nagy, and D. P. O'Leary. *Deblurring Images: Matrices, Spectra, and Filtering (Fundamentals of Algorithms 3) (Fundamentals of Algorithms)*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2006.

[25] C. Harris and M. Stephens. A combined corner and edge detector. In *Fourth Alvey Vision Conference*, pages 147–151, 1988.

[26] J. Jia. Single image motion deblurring using transparency. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[27] H. Jin, P. Favaro, and R. Cipolla. Visual tracking in the presence of motion blur. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18–25. IEEE Computer Society, 2005.

[28] S. Kang, J. Min, and J. Paik. Segmentation-based spatially adaptive motion blur removal and its application to surveillance systems. In *International Conference on Image Processing, (ICIP)*, pages I: 245–248, 2001.

[29] G. Klein and T. Drummond. Tightly integrated sensor fusion for robust visual tracking. *Image and Vision Computing*, 22(10):769–776, September 2004.

[30] G. Klein and T. Drummond. A single-frame visual gyroscope. In *British Machine Vision Conference*, pages xx–yy, 2005.

[31] C. Kolb, D. Mitchell, and P. Hanrahan. A realistic camera model for computer graphics. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 317–324, New York, NY, USA, 1995. ACM.

[32] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing. School of Computer Science & Software Engineering, The University of Western Australia. Available from: <http://www.csse.uwa.edu.au/~pk/research/matlabfns/>.

[33] A. Levin. Blind motion deblurring using image statistics. In *Advances in Neural Information Processing Systems (NIPS)*, pages 841–848, 2006.

[34] A. Levin, R. Fergus, F. Durand, and W. T. Freeman. Image and depth from a conventional camera with a coded aperture. *ACM Transactions on Graphics*, 26(3):70:1–70:9, July 2007.

[35] A. Levin, A. Rav-Acha, and D. Lischinski. Spectral matting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(10):1699–1712, 2008.

[36] F. Li, J. Yu, and J. Chai. A hybrid camera for motion deblurring and depth map super-resolution. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008.

[37] H. Lin and C. Chang. Photo-consistent motion blur modeling for realistic image synthesis. In *Pacific Rim Symposium on Image and Video Technology (PRSIVT 2006)*, LECTURE NOTES IN COMPUTER SCIENCE, pages 1273–1282. SPRINGER-VERLAG BERLIN, 2006.

[38] D. G. Lowe. Object recognition from local scale-invariant features. In *ICCV '99: Proceedings of the International Conference on Computer Vision-Volume 2*, page 1150, Washington, DC, USA, 1999. IEEE Computer Society.

[39] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004.

[40] L. Lucy. An iterative technique for the rectification of observed distributions. *Astronomy J.*, 79:745–754, 1974.

[41] B. M. and L. D. G. Invariant features from interest point groups. In *British Machine Vision Conference, Cardiff, Wales*, pages 656–665, 2002.

[42] C. Mei and I. Reid. Modeling and generating complex motion blur for real-time tracking. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, 2008.

[43] K. Mikolajczyk and C. Schmid. Aperformance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:1615–1630, October 2005.

[44] H. P. Moravec. *Obstacle avoidance and navigation in the real world by a seeing robot rover*. PhD thesis, Stanford University, Stanford, CA, USA, 1980.

[45] C. Z. Z. K. P. R. J. O.-M. J. Z. N. Dey, L. Blanc-Fraud. A deconvolution method for confocal microscopy with total variation regularization. In *IEEE International Symposium on Biomedical Imaging: Nano to Macro*, 2004.

[46] M. Potmesil and I. Chakravarty. Modeling motion blur in computer-generated images. *SIGGRAPH Comput. Graph.*, 17(3):389–399, 1983.

[47] A. Pretto, E. Menegatti, M. Bennewitz, W. Burgard, and E. Pagello. A visual odometry framework robust to motion blur. In *ICRA'09: Proceedings of the 2009 IEEE international conference on Robotics and Automation*, pages 1685–1692, Piscataway, NJ, USA, 2009. IEEE Press.

[48] R. Raskar, A. K. Agrawal, and J. Tumblin. Coded exposure photography: motion deblurring using fluttered shutter. *ACM Trans. Graph.*, 25(3):795–804, 2006.

[49] A. Rav-Acha and S. Peleg. Two motion-blurred images are better than one. *Pattern Recogn. Lett.*, 26(3):311–317, 2005.

[50] W. Richardson. Bayesian-based iterative method of image restoration. *Journal of the Optical Society of America*, 62(1):55–59, January 1972.

[51] C. Schmid, R. Mohr, and C. Bauckhage. Evaluation of interest point detectors. *International Journal of Computer Vision*, 37(2):151–172, 2000.

[52] Q. Shan, J. Jia, and A. Agarwala. High-quality motion deblurring from a single image. *ACM Transactions on Graphics (SIGGRAPH)*, 27:1–10, 2008.

[53] Q. Shan, W. Xiong, and J. Jia. Rotational motion deblurring of a rigid object from a single image. In *IEEE 11th International Conference on In Computer Vision*, pages 1–8, 2007.

[54] M. Sorel and J. Flusser. Blind restoration of images blurred by complex camera motion and simultaneous recovery of 3d scene structure. In *5th IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), Vols 1 and 2*, pages 737–742, 2005.

[55] M. Sorel and J. Flusser. Space-variant restoration of images degraded by camera motion blur. *IEEE Transactions on Image Procecessing*, 17(2):105–116, February 2008.

[56] H. Stark and J. W. Woods. *Probability, random processes, and estimation theory for engineers*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1986.

[57] J. Telleen, A. Sullivan, J. Yee, O. Wang, P. Gunawardane, I. Collins, and J. Davis. Synthetic shutter speed imaging. *Comput. Graph. Forum*, 26(3):591–598, 2007.

[58] D. L. Tull and A. K. Katsaggelos. Regularized blur-assisted displacement field estimation. In *IEEE International Conferance on Image Processing*, pages III: 85–88, 1996.

[59] Y. F. Wang and P. Liang. 3d shape and motion analysis from image blur and smear: A unified approach. *Computer Vision, IEEE International Conference on*, 0:1029, 1998.

[60] C. Webster and S. Reeves. Radial deblurring with ffts. pages I: 101–104, 2007.

[61] W. Wenying, H. Jingxin, and Y. Yi. Identification of blurred extent with rotation motion blurred image. In *ISISE '08: Proceedings of the 2008 International Symposium on Information Science and Engieering*, pages 669–672, Washington, DC, USA, 2008. IEEE Computer Society.

[62] N. Wiener. *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*. The MIT Press, 1964.

[63] M. M. Wloka and R. C. Zeleznik. Interactive real-time motion blur. *The Visual Computer*, 12:283–295, 1996.

[64] Y. Yokokohji, Y. Sugawara, and T. Yoshikawa. Accurate image overlay on video see-through hmds using vision and accelerometers. In *IEEE Virtual Reality Conference*, page 247, Washington, DC, USA, 2000. IEEE Computer Society.

[65] S. You and U. Neumann. Fusion of vision and gyro tracking for robust augmented reality registration. In *Proceedings of the Virtual Reality Conference.IEEE*, page 71, Washington, DC, USA, 2001. IEEE Computer Society.

[66] L. Yuan, J. Sun, L. Quan, and H.-Y. Shum. Image deblurring with blurred/noisy image pairs. In *ACM SIGGRAPH*, page 1, New York, NY, USA, 2007. ACM.

[67] L. Yuan, J. Sun, L. Quan, and H.-Y. Shum. Progressive inter-scale and intra-scale non-blind image deconvolution. *ACM Trans. Graph.*, 27(3):1–10, 2008.

APPENDIX A

FRAMES OF THE TEST VIDEO SEQUENCES

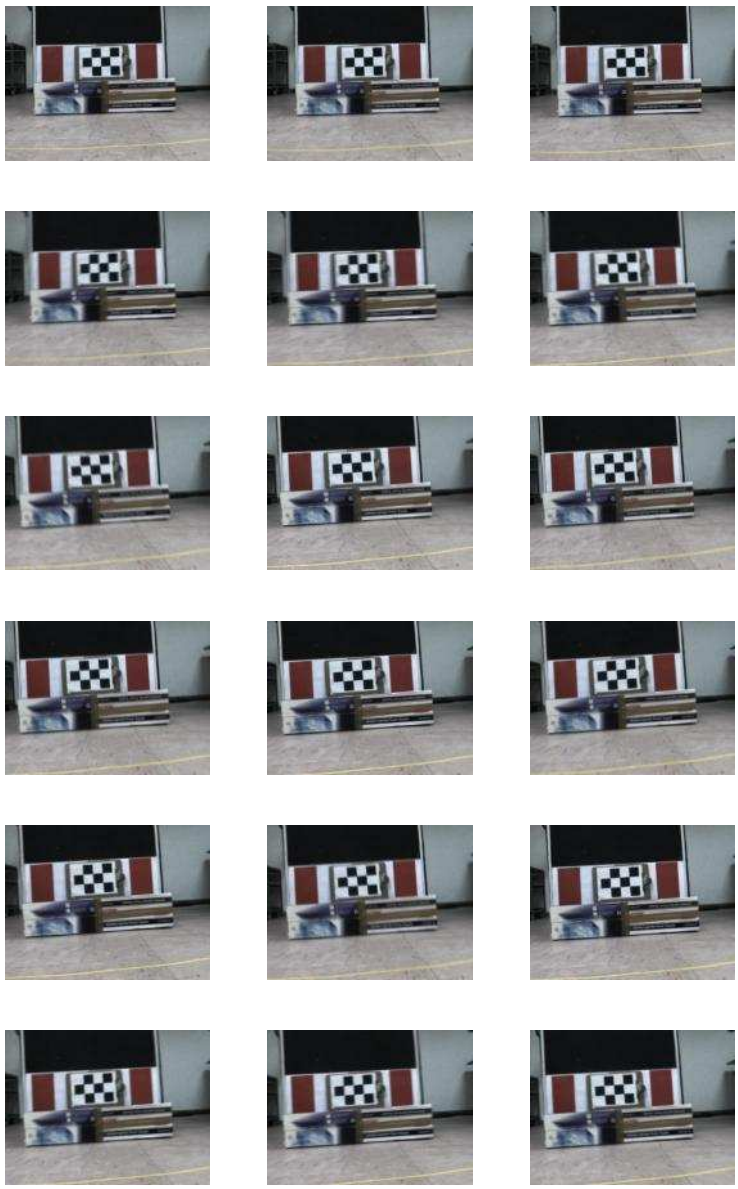


Figure A.1: 1025 to 1042 frames of the test video recorded while our experimental robotic platform RHex was walking with 0.1 velocity coefficient.

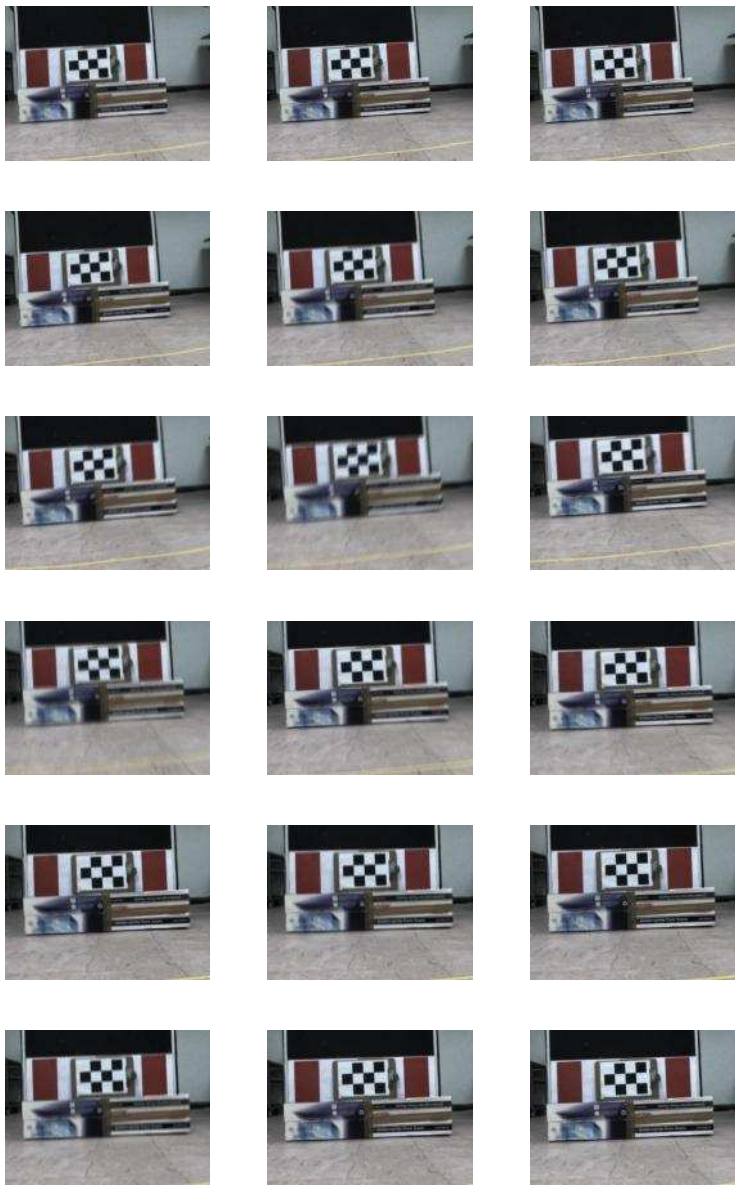


Figure A.2: 1043 to 1060 frames of the test video recorded while our experimental robotic platform RHex was walking with 0.1 velocity coefficient.

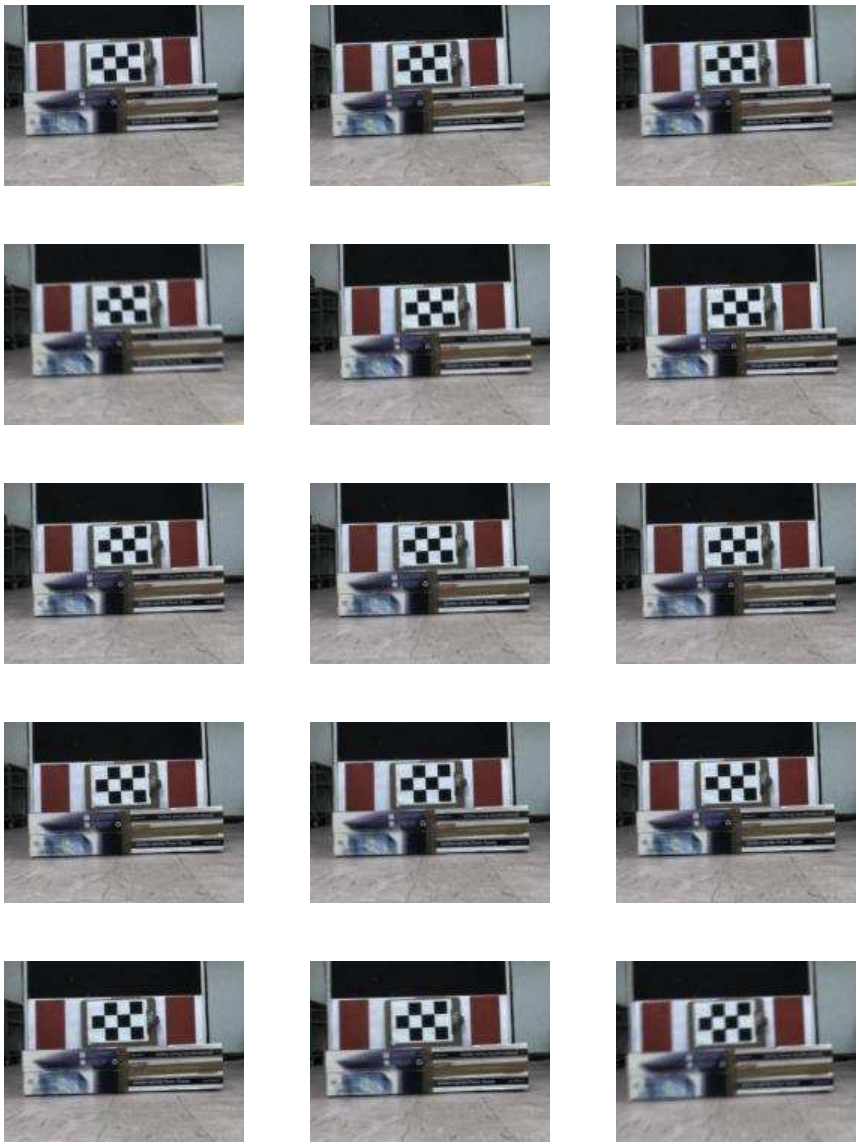


Figure A.3: 1061 to 1075 frames of the test video recorded while our experimental robotic platform RHex was walking with 0.4 velocity coefficient.

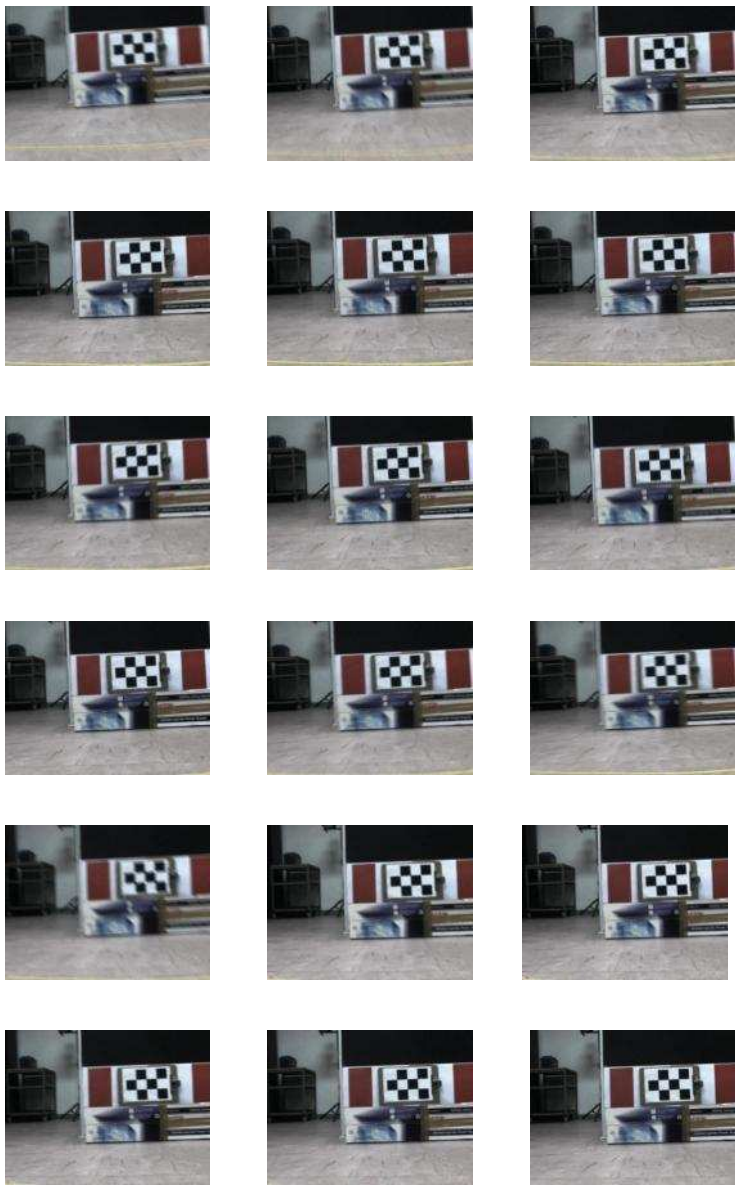


Figure A.4: 1425 to 1442 frames of the test video recorded while our experimental robotic platform RHex was walking with 0.4 velocity coefficient.

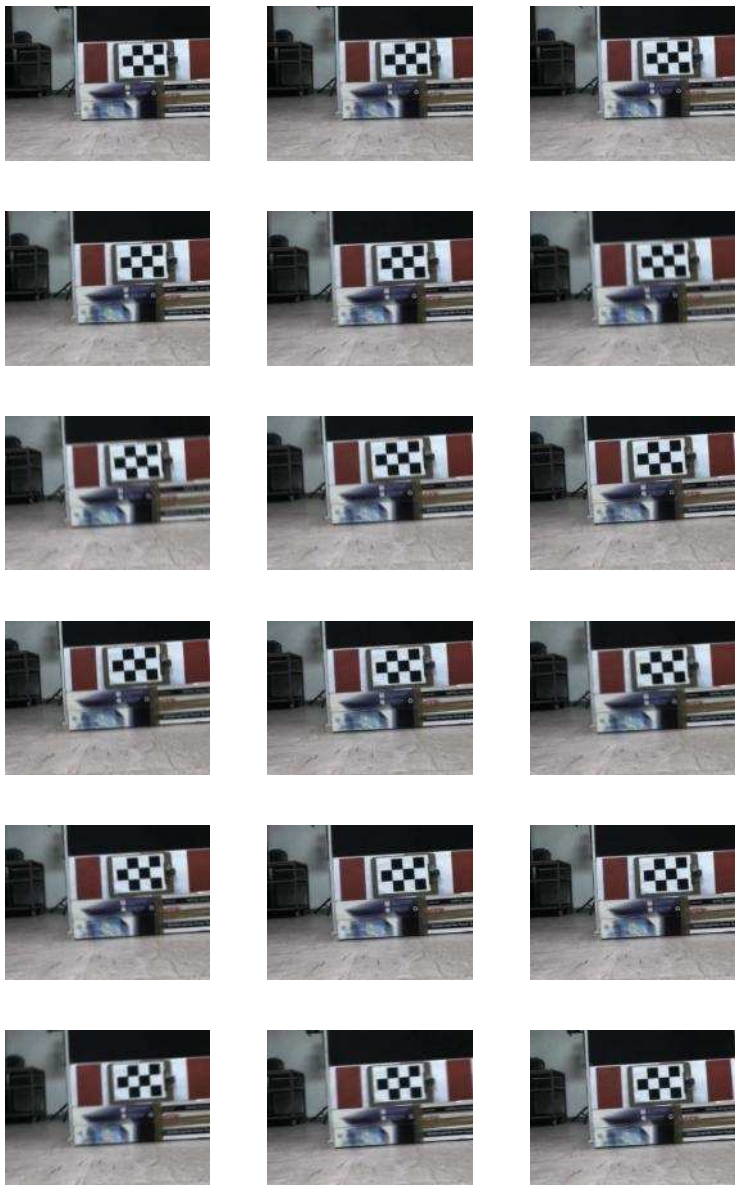


Figure A.5: 1443 to 1460 frames of the test video recorded while our experimental robotic platform RHex was walking with 0.4 velocity coefficient.

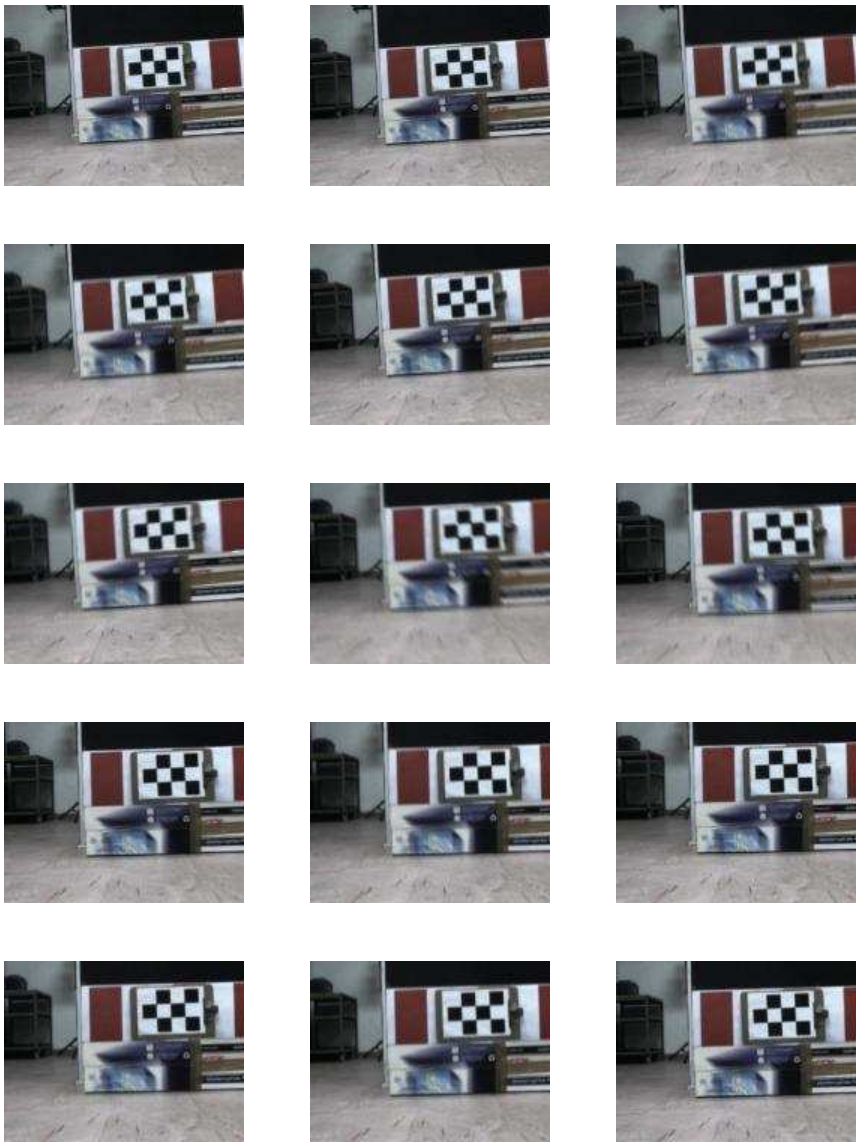


Figure A.6: 1461 to 1475 frames of the test video recorded while our experimental robotic platform RHex was walking with 0.4 velocity coefficient.

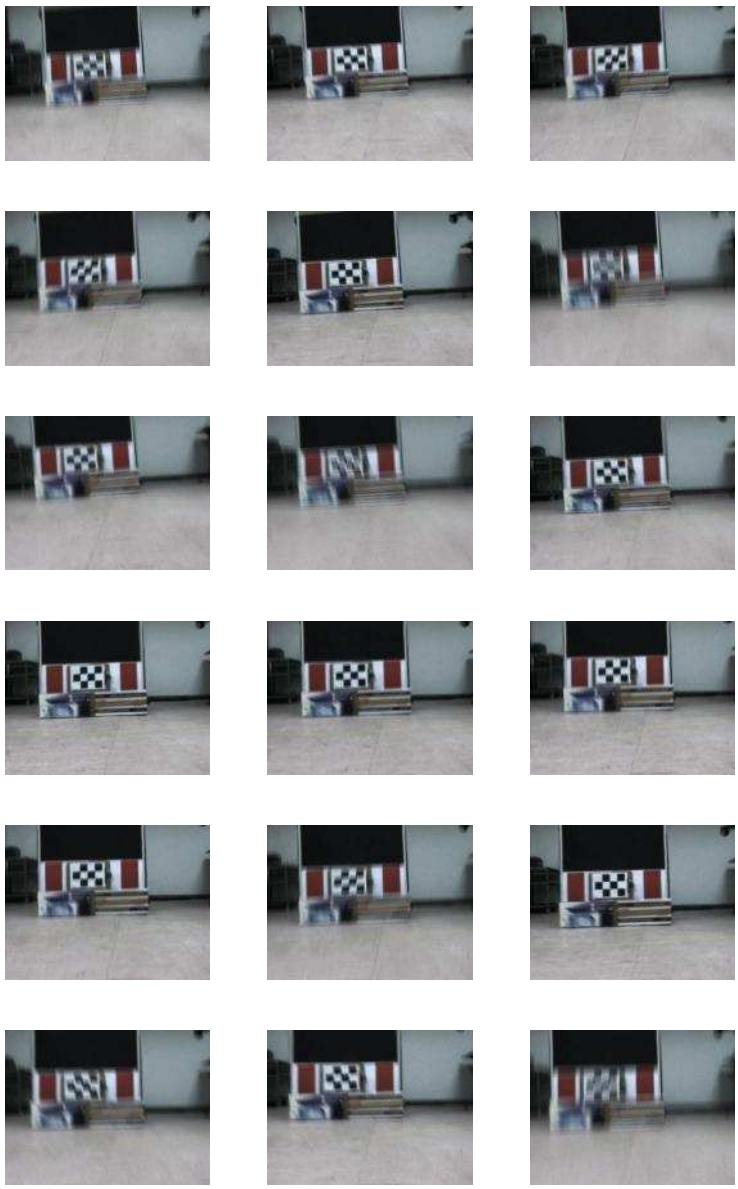


Figure A.7: 725 to 742 frames of the test video recorded while our experimental robotic platform RHex was walking with 0.8 velocity coefficient.

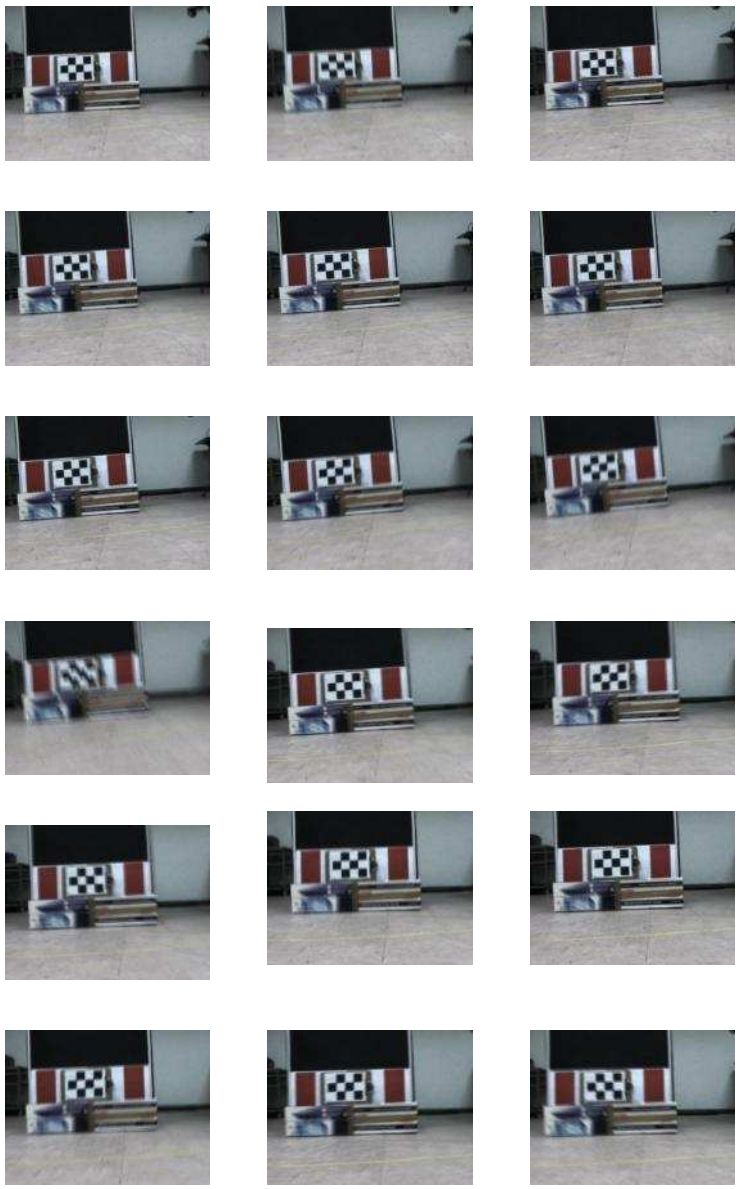


Figure A.8: 743 to 760 frames of the test video recorded while our experimental robotic platform RHex was walking with 0.8 velocity coefficient.

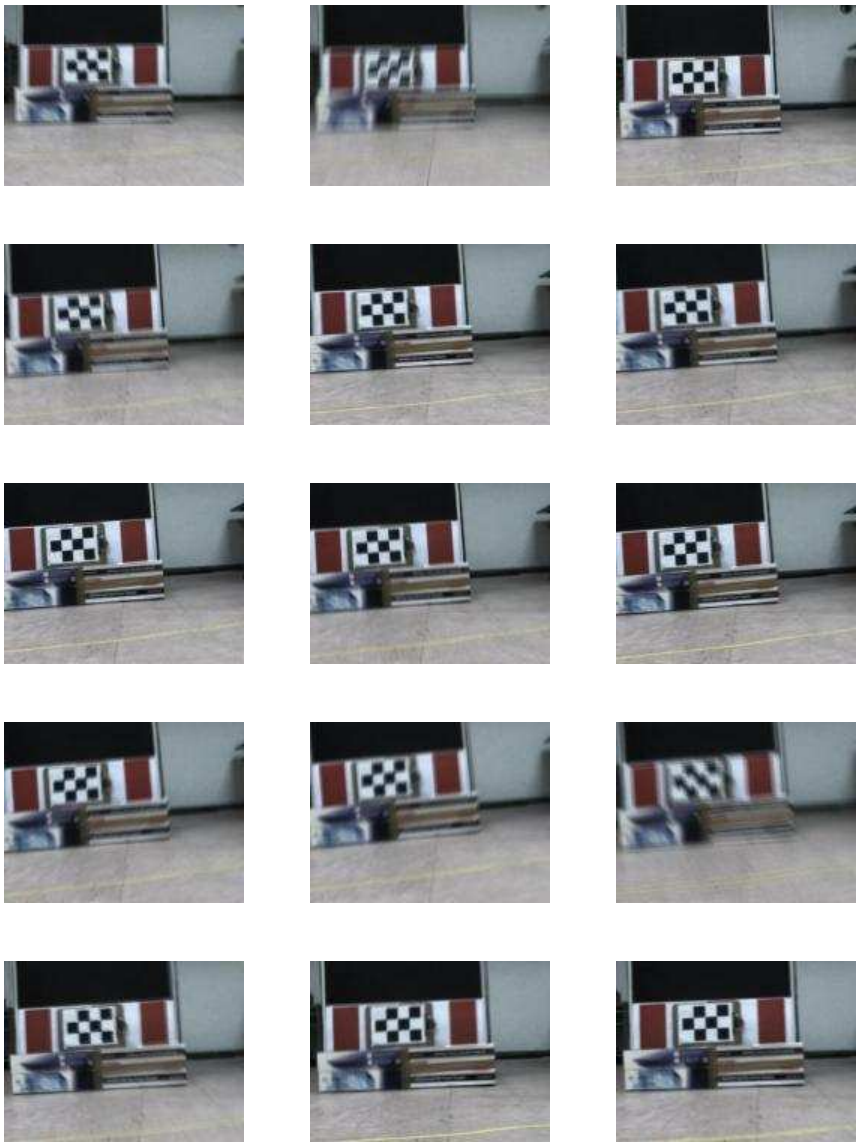


Figure A.9: 761 to 775 frames of the test video recorded while our experimental robotic platform RHex was walking with 0.8 velocity coefficient.

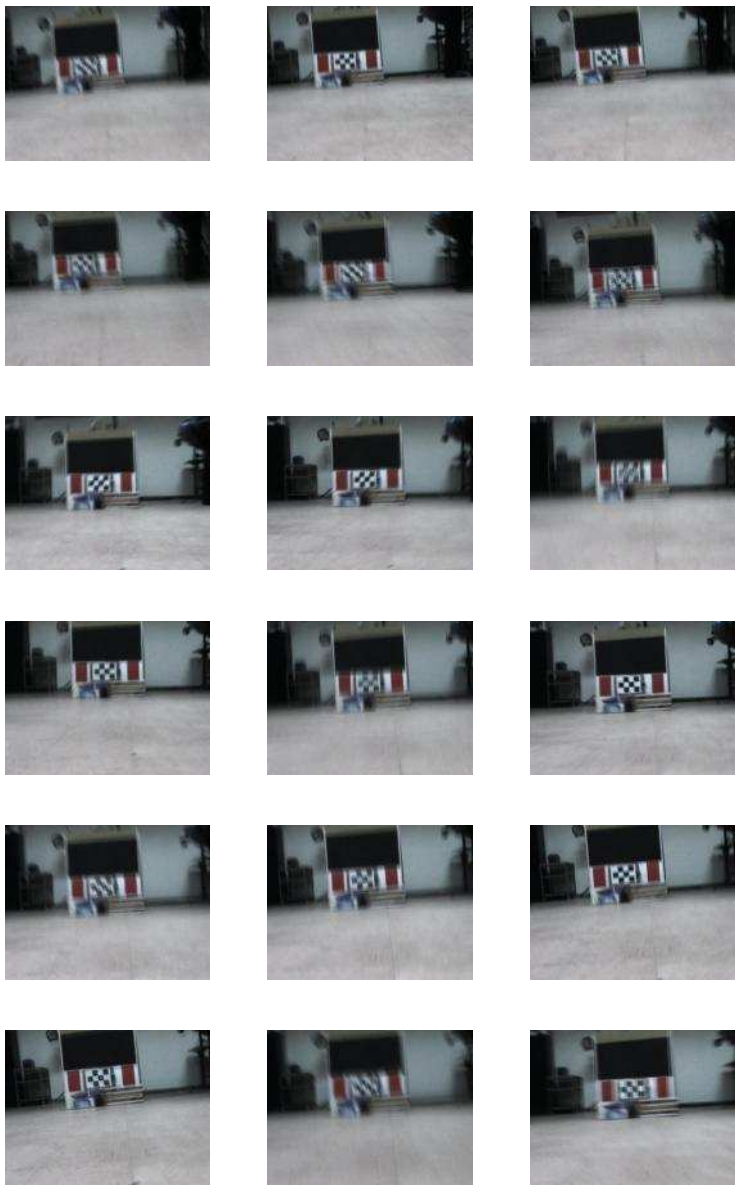


Figure A.10: 440 to 457 frames of the test video recorded while our experimental robotic platform RHex was walking with 1.0 velocity coefficient.

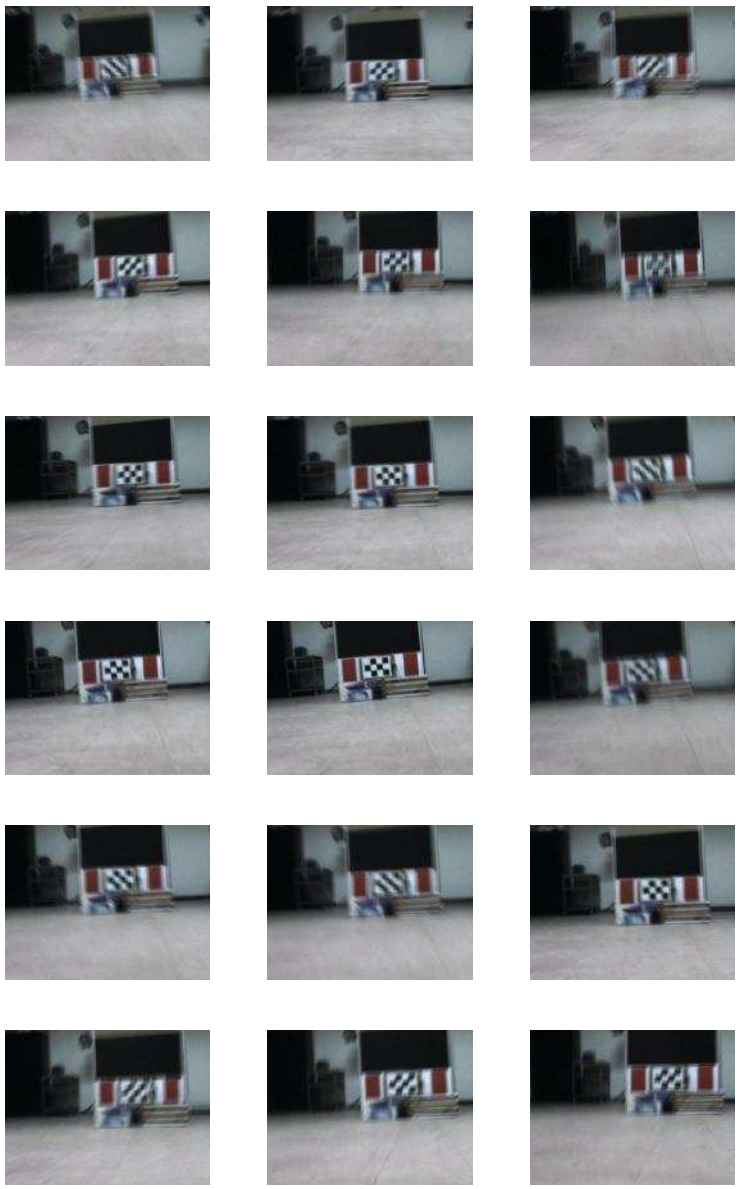


Figure A.11: 458 to 475 frames of the test video recorded while our experimental robotic platform RHex was walking with 1.0 velocity coefficient.

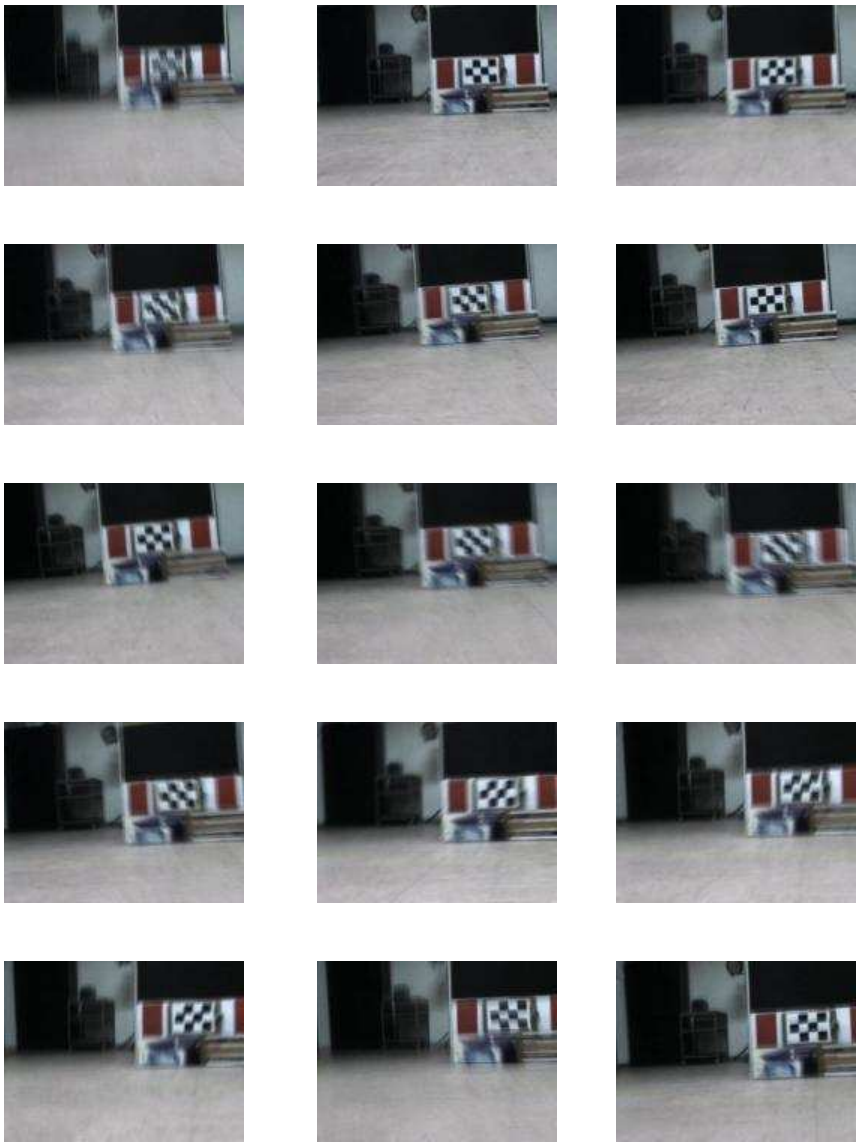


Figure A.12: 476 to 490 frames of the test video recorded while our experimental robotic platform RHex was walking with 1.0 velocity coefficient.