Robotic Riding Mechanism for Segway Personal Transporter

WONG, Sheung Man

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Master of Philosophy
in
Automation and Computer-Aided Engineering

The Chinese University of Hong Kong
April 2010

Thesis Committee

Professor Chung, Chi-kit, Ronald (Chair)

Professor Yam, Yeung (Thesis Supervisor)

Professor Liu, Yunhui (Committee Member)

Professor Sun, Dong (External Examiner)

Abstract

Robotic car driver is commonly developed and well established. Many researches are working on this issue and also on value-added functions, such as auto-piloting, self-navigation and localization based on the robotic driver. Researches on robotic driver can be found whenever a new moving platform is developed.

Segway Personal Transporter (PT) is a recently developed two-wheel transporter. The characteristic of Segway PT is the two-wheel driving system across the same axis. The system is an inverted pendulum and is potentially unstable. It needs self-providing force to balance whole system.

This thesis presents the design, dynamic models and approaches of constructing a robotic riding platform for maneuvering a Segway Personal Transporter. The thesis includes the designs and construction of the hardware as well as dynamic model and control theory applicable to the problem. A servo motor driven platform is developed to control position of center of mass of the system. Infrared sensors, ultrasonic sensors and a webcam are also integrated on our system to achieve auto-piloting function.

Other than the constructed platform, several applications from similar platforms, for example, auto-piloting, self-navigation, localization and 3D environment modeling, are also studied. Based on their approaches, tailor-made auto-piloting method for the Segway Personal Transporter based platform is developed. Studies on other potential applications of the platform are also introduced.

Vision system is used as the main media for auto-piloting of our system. The vision process involves several image processing methods such as color space selection, edge detection and Hough transform.

An experiment, termed the Grand Challenge, is finished on 27 April, 2009 at the CUHK University Gymnasium running track to validate the performance of system. Time needed for completing a loop on the track was five minutes and twenty-five seconds. The Challenge was successfully finished on same running lane without crossing the lane border for the whole journey.

摘要

近年，機器人駕駛員技術不斷發展。除了機器人駕駛之核心技術研究，不少研究專注於其他增值功能，例如自動導航和自動定位等。每當有新移動平台面世，都會有很多建基於新平台的機器人駕駛研究出現。

Segway 個人代步工具（PT）是一種新發展的兩輪交通工具。其特色在於兩輪是置於同軸線上。整個系統屬於潛在不穩定結構及倒鐘擺式設計，因此，整個系統需不斷移動以維持其自身平衡。

本論文敘述建基於 Segway PT 的機器人騎士平台的設計與製造過程。由於本項目涉及硬件製造，本文會描述硬件的設計概念和有關之控制理論。我們開發之系統中包含一個以伺服馬達驅動之平台，用以控制整個系統的位置。另外，爲了建立自動導航功能，我們爲系統加上紅外線感測器、超聲波感測器和網絡攝影機。

在研究過程中，我們研讀其他相類似平台的著作，包括自動導航、自動定位和三維環境掃瞄等。參考這些研究，本項目開發出爲 Segway PT 度身訂造的自動導航技術。另外，本文亦探討了未來可建基於本平台的研究項目。

在整個系統中，視像系統被用作自動導航的主要感測媒介。因此，色域選取、邊緣檢測和霍夫變換等影像處理技術被應用於系統之中。

爲証實和檢測系統的表現，我們於二零零九年四月二十七日在中文大學夏鼎基運動場跑道完成了實地試驗。我們的機器人成功以全自動和全程不橫越起始跑道線的情況下，以五分二十五秒完成一圈（四百米）的路程。

Acknowledgements

I would like to express my sincere thanks to my supervisor, Professor Yam Yeung, for all his guidance, support and invaluable advices during my study. Thank for giving me the chance to learn the attitude of working on researches.

Besides, I would like to thanks all colleagues in the Intelligent Control Systems Laboratory for their support and advice. Special thanks are given to K.W. Lo and K.W. Kwok for their leading to the research road.

Finally, I would like to thank my family for their patient and support.

List of figures

Table of Contents

Chapter 1

Introduction

1.1. Segway Personal Transporter (PT)

Mobility robotic platform based on concept of inverted pendulum attracted lots of research interest in recent years [1]. Dynamic modeling for such and similar designs are well-conducted. The interest of researchers are moving to the development of application using such systems that feature the desirable characteristics of zero-turning radius, smaller in size and easy to control.

Segway is invented by Dean Kamen. It is a two-wheeled, self-balancing electric vehicle which was unveiled in 2001 and first produced as Segway Personal Transporter (PT for short in follows, previously named as Human Transporter – HT. Fig.1.1) in 2002 [1]. Segway PT is the first commercialized electric vehicle using mechanical design concept of inverted pendulum. The Segway is self-balanced by its internal system if sufficient counter-balance on load.

The working principle of Segway is controlling the position of center of mass (CM) of the system. When the center of mass moves forward, the motors of Segway accelerates towards same direction in order to maintain the balance of the system. Using this characteristic, this project aims to develop and attach a mechanical platform on the Segway PT which controls the system by changing the position of the CM of the platform.

Figure 1.1. Segway PT (i series)

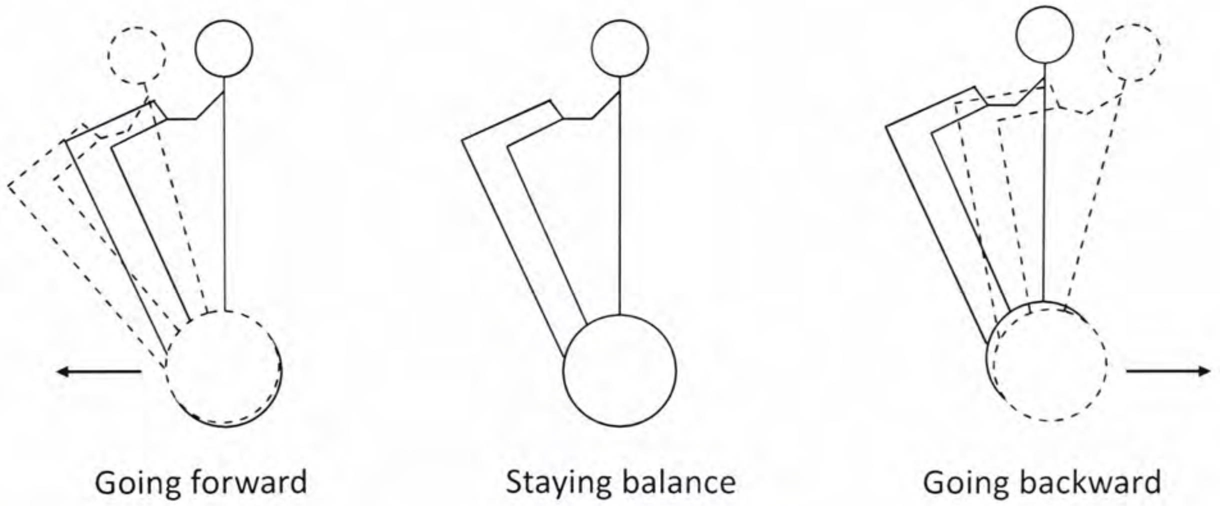

Figure 1.2. Demonstrating the balancing of Segway

The figure 1.2 shows that the motor of Segway will follow the center of mass in order to balance the whole system. The more the CM shifts from the balancing point, the system has larger velocity and acceleration towards that direction.

1.2. Existing research using Segway Robotic Mobility Platform™ (RMP)

There were many institutes developed robots using Segway RMP as their research platform, including NASA [4][5], MIT [6], CMU [7], NSI [8], Stanford [9], etc. These researches develop different applications using Segway RMP to replace traditional mobility platform.

Based on Segway PT, Segway Robotic Mobility Platform™ (RMP) is developed as a new mobile robotic platform which is shown on figure 1.3. The RMP is equipped with all features that Segway PT has, i.e. small footprint, zero turning radius, and large payload, etc. In addition, users of Segway RMP can directly control the motion of the RMP through laptop mounted on the platform. For this reason, most researchers chose Segway RMP as their development platform for their mobile platform projects [3].

Figure 1.3. The Segway RMP

NASA used Segway RMP to develop a low-cost, earth-based Robonaut for studying coordinated control of dexterous limbs on a mobile robot (Figure 1.4) [4][5]. The Robonaut body allows 45 DOF motion for simulating human motions. It can be remotely controlled by telepresence hardware. The operator can control the Robonaut through 3D motion tracking system. The tracking system can capture operator's body motion and send control commands through wireless transmission system. As the system is remotely controlled, stereo video from the Robonaut feed to operator's VR helmet in order to provide live video view of the Robonaut.

Figure 1.4. The NASA Robonaut RMP

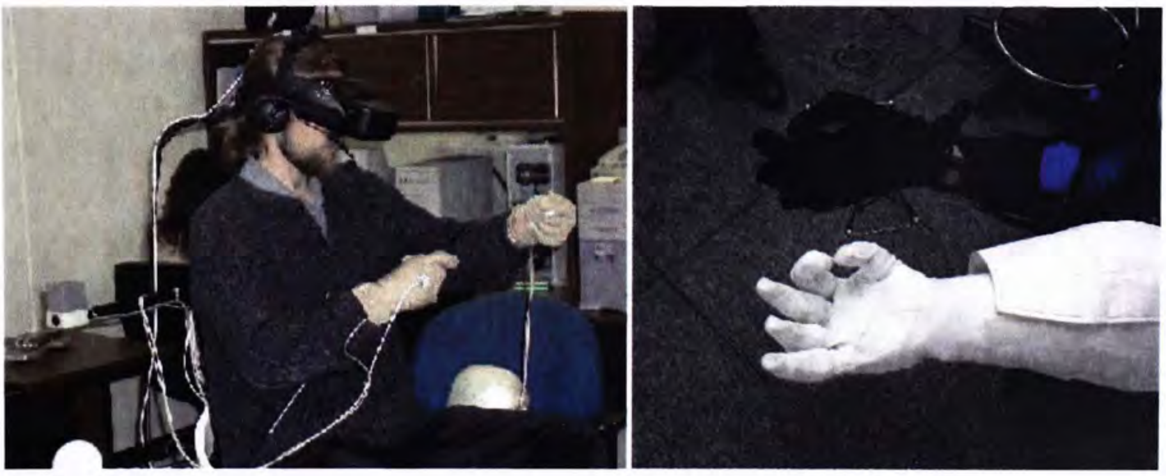

Figure 1.5(a) (Left). Telepresence Hardware

Figure 1.5(b) (Right). Cartesian control using prototype optical glove

MIT equipped a single 5-DOF arm on the Segway RMP for navigating indoor office environment and named it as Cardea (Figure 1.6) [6]. It uses vision system and ultrasonic sensors for indoor navigation. This project demonstrates possibility on combining indoor auto-navigation and robot arm.

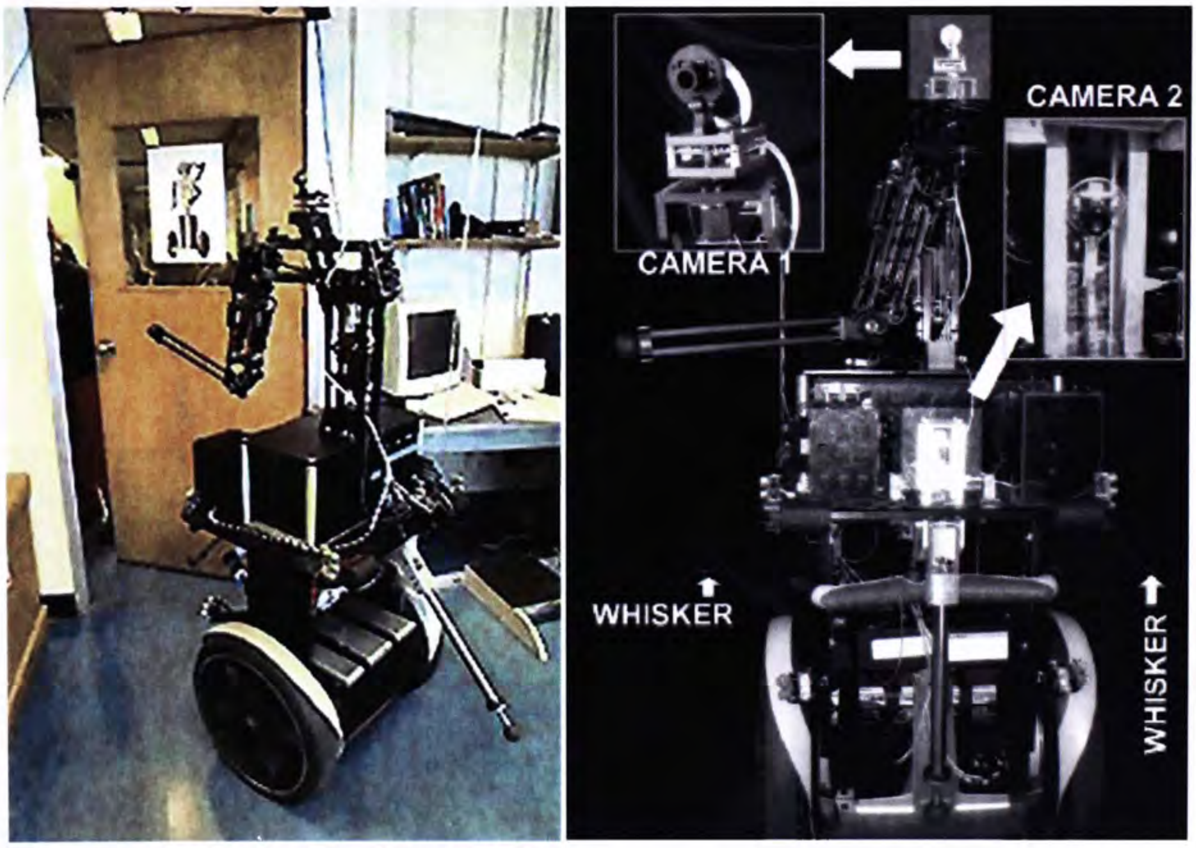

Figure 1.6. MIT's Cardea

The Cardea uses image processing method for indoor floor features searching, door position searching and door handle detection. Color segmentation plays an important role in its image processing progress.

In its floor detection process (Figure 1.7), the system scans vertical columns of pixel. Position of edges between floor and obstacles is determined by comparing color inside columns. Direction for robot movement and distribution of obstacles can be found.

Figure 1.7. MIT's Cardea: example on floor detection

Both door detection (Figure 1.8) and door handle detection (Figure 1.9) process for Cardea adopted Hough transform to determine line features in the images. Based on line information extracted from the images, doors can be recognized by the robot and door handle position can be provided to its robot arm for door opening motion.

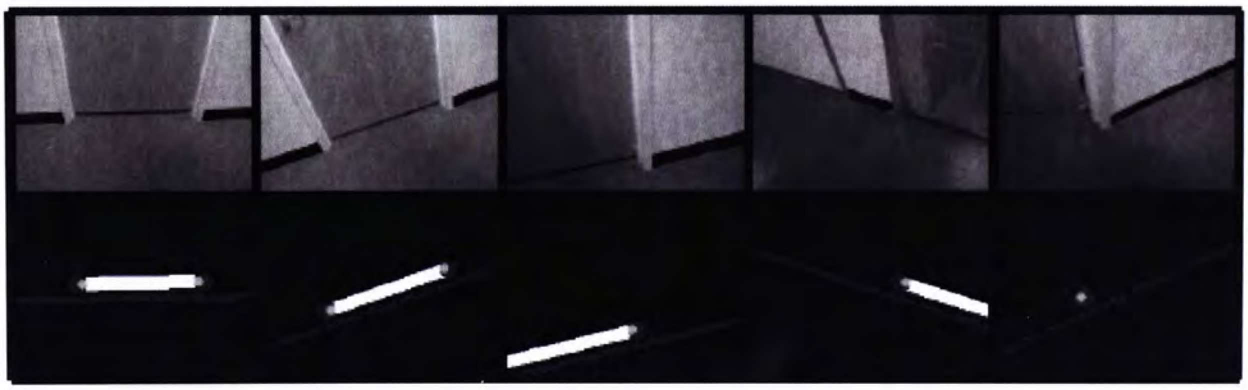

Figure 1.8. MIT's Cardea: example on door detection

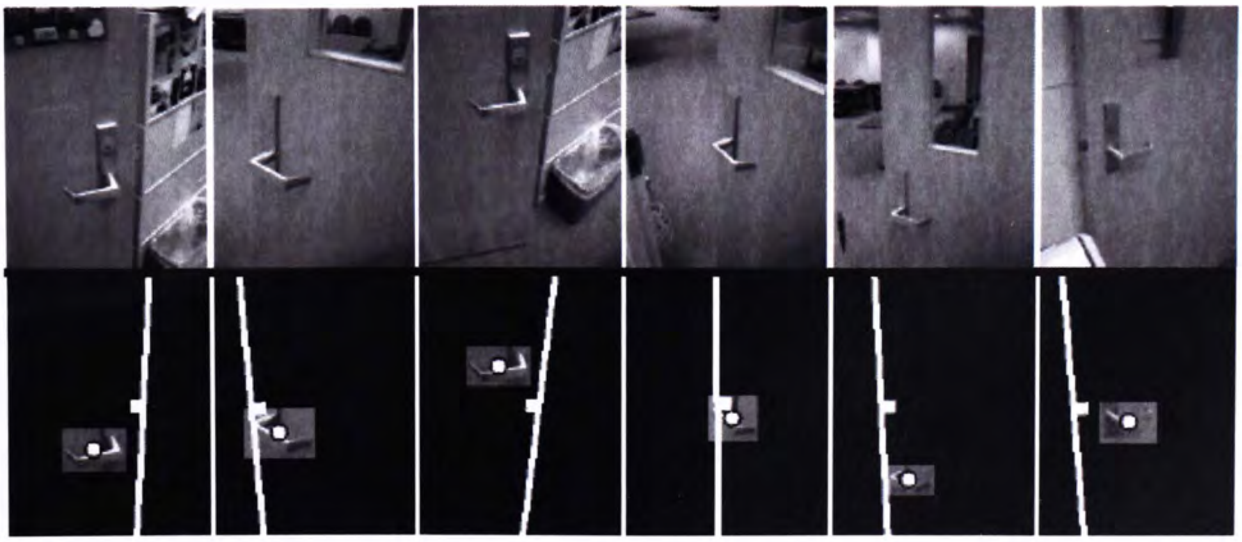

Figure 1.9. MIT's Cardea: example on door handle detection

CMU built a Segway Soccer which is equipped with camera and communication ability (Figure 1.10) [7][13]. This project demonstrates possibilities on reliable object tracking and interactions between robots.

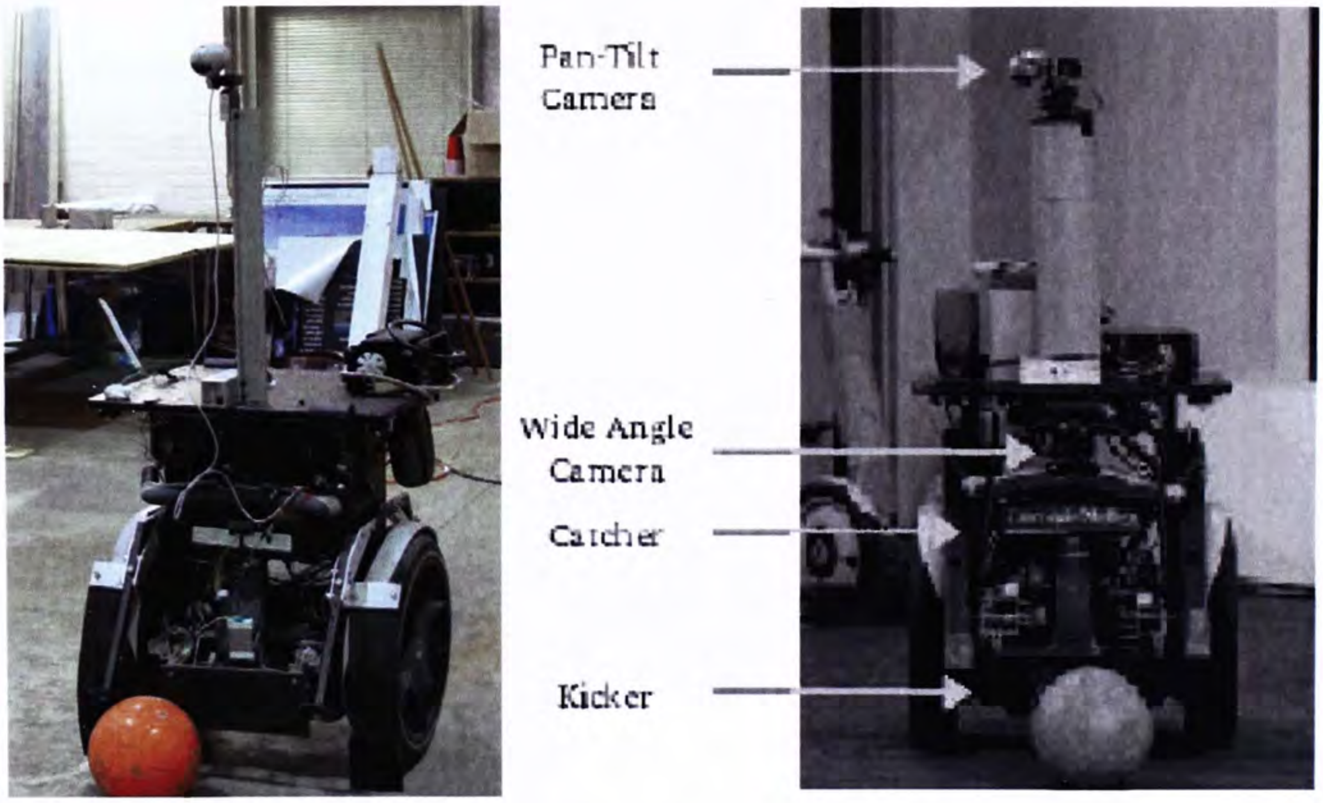

Figure 1.10. CMU's Segway Soccer

The Segway Soccer adopted color object recognition (Figure 1.11) on its vision system to search position of ball and goals. It also allows team play between robot-robot, human-robot and human-human teams.

Figure 1.11. CMU's Segway Soccer: example on its vision system

Pink crosses represent goals; Blue points represent estimated position of ball

Stanford University built a Segway RMP based mobile robot for outdoor terrain scanning and modeling [9]. The robot is equipped with SICK laser range finder for 3D scanning and GPS receiver for localization. It shows usage of Segway-liked platform on large-scale outdoor applications.

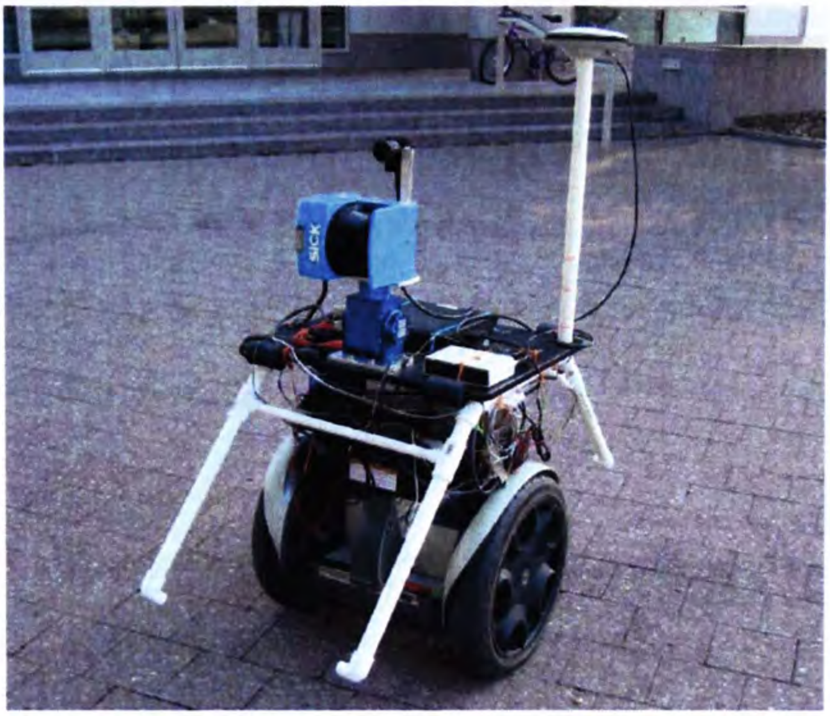

Figure 1.12.Stanford University's mobile robot

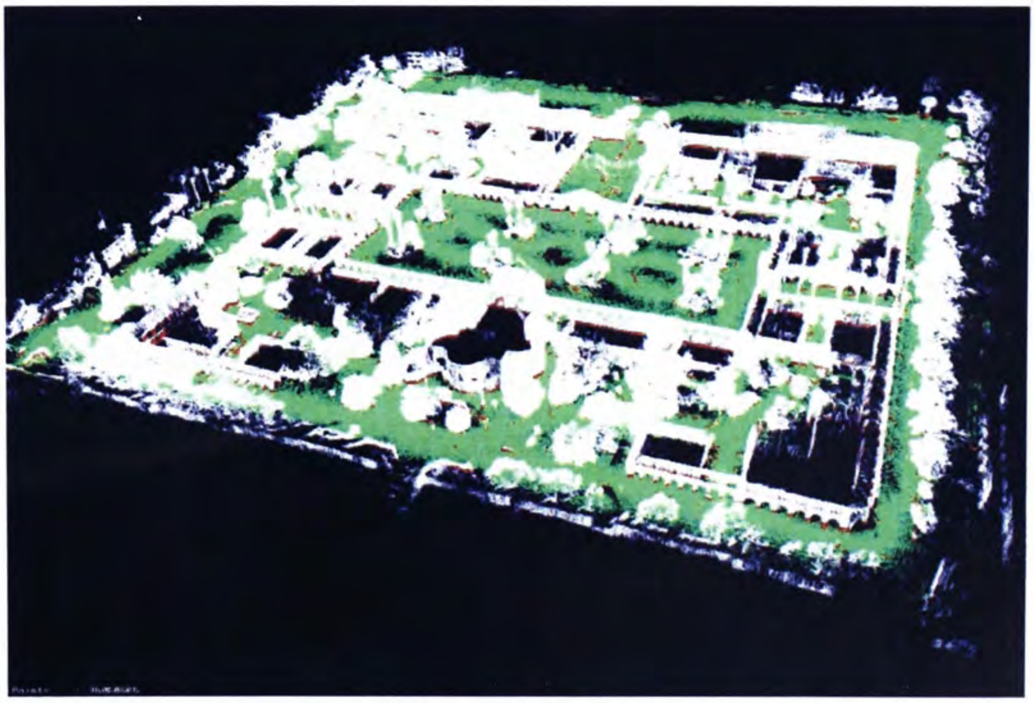

Figure 1.13. 3-D map of the center of the Stanford campus constructed by the mobile robot

1.3. The ICSL Segway Rider

In our Intelligent Control Systems Laboratory (ICSL), a Segway Robotic Rider is constructed based on Segway PT i-series. There are two reasons for choosing Segway PT as our research platform instead of Segway RMP.

Firstly, Segway RMP can be controlled by laptop or any other computer directly through CAN interface. That means Segway RMP can be used similar to traditional mobility platform by giving commands directly. It is difficult to develop a human-like robotic platform to simulate human-Segway interactions.

Unlike Segway RMP, Segway PT can only be controlled by external physical movement of the CM. Our design of Segway Rider is equipped with a controlled moving mass built on the Rider platform. By changing the position of the mass, the CM of entire system is changed. The Segway PT can respond to the changes automatically to balance the entire system. Therefore, human-Segway interactions can be partially simulated through our Rider.

In addition, since our Rider is completely independent system with the Segway without any electrical and electronic connections with it. The Rider can be used for all models of Segway PT by making slight modification on the mounting of the Rider.

Cost is another reason for choosing Segway PT instead of Segway RMP. Segway PT costs around US$5000 while Segway RMP costs US$17000 which is a substantial difference.

1.4. Thesis outlines

In what follows, Chapter 2 describes the hardware of the Segway Rider in detail. Description on software development for the Rider, including user interface and program data flow, is also included. Chapter 3 describes the grand challenge for the Segway Rider. The required computer vision algorithm implemented on the Rider is explained in detail. The criteria of choosing image processing algorithm are also discussed. The "Stand and Stay" function of the Rider is described in Chapter 4. The Rider can keep itself within a relatively narrow area. It is an important function for robotics mobility platform. Finally, the conclusions are given in Chapter 5. Possible future research directions are also discussed

Chapter 2

ICSL Segway Rider

2.1. Design concept

The objective of this project is constructing a mobile platform that is able to control Segway PT using external mechanical parts without changing any original Segway setting.

To achieve the target stated above, our platform contains a horizontal moving mechanical part to change the CM of the system. If the changes of CM are large enough, the system can drive the Segway forward and backward. By controlling the scale of change of center of mass, velocity and acceleration of the system can also be controlled.

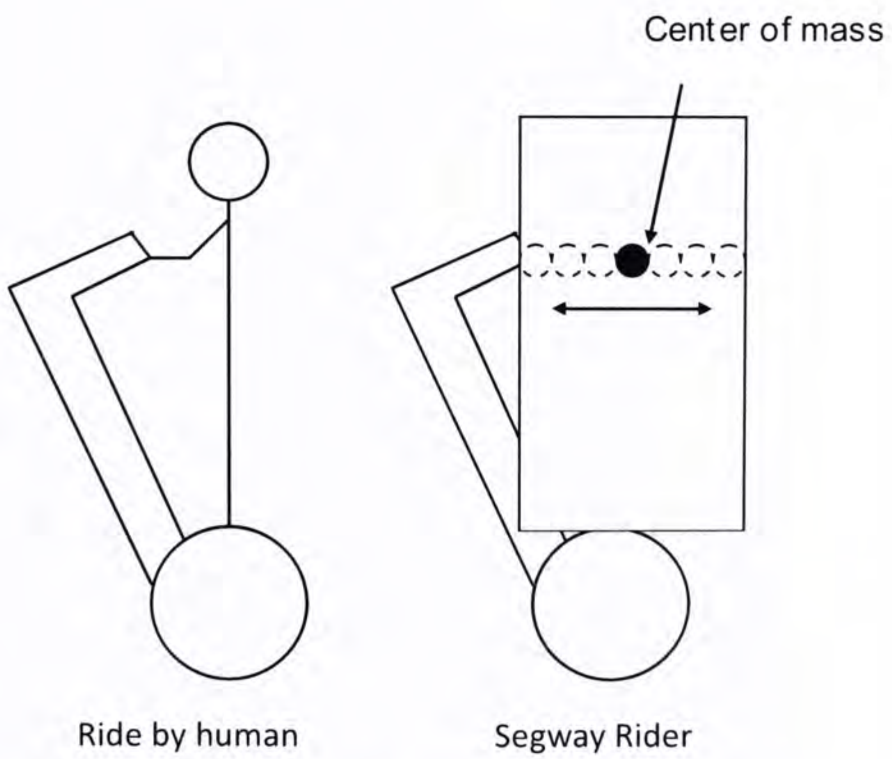

Figure 2.1. Design concept of Segway Rider

2.2. Design overview

The ICSL Segway Rider is attached on a Segway PT i-series as mobility platform. The structure of the Rider is built by aluminum profile. The Rider is equipped with a linear screw for moving the CM of the entire system and a high torque metal gear servo-motor for controlling turning switch of the Segway PT. The system is designed for conducting research of different applications on a well-developed mobility platform. Both hardware and software space for adding new sensors and equipments is reserved for future developments. The overview hardware of the Rider is shown in Figure 2.2.

Figure 2.2. Segway Rider mounted on Segway PT

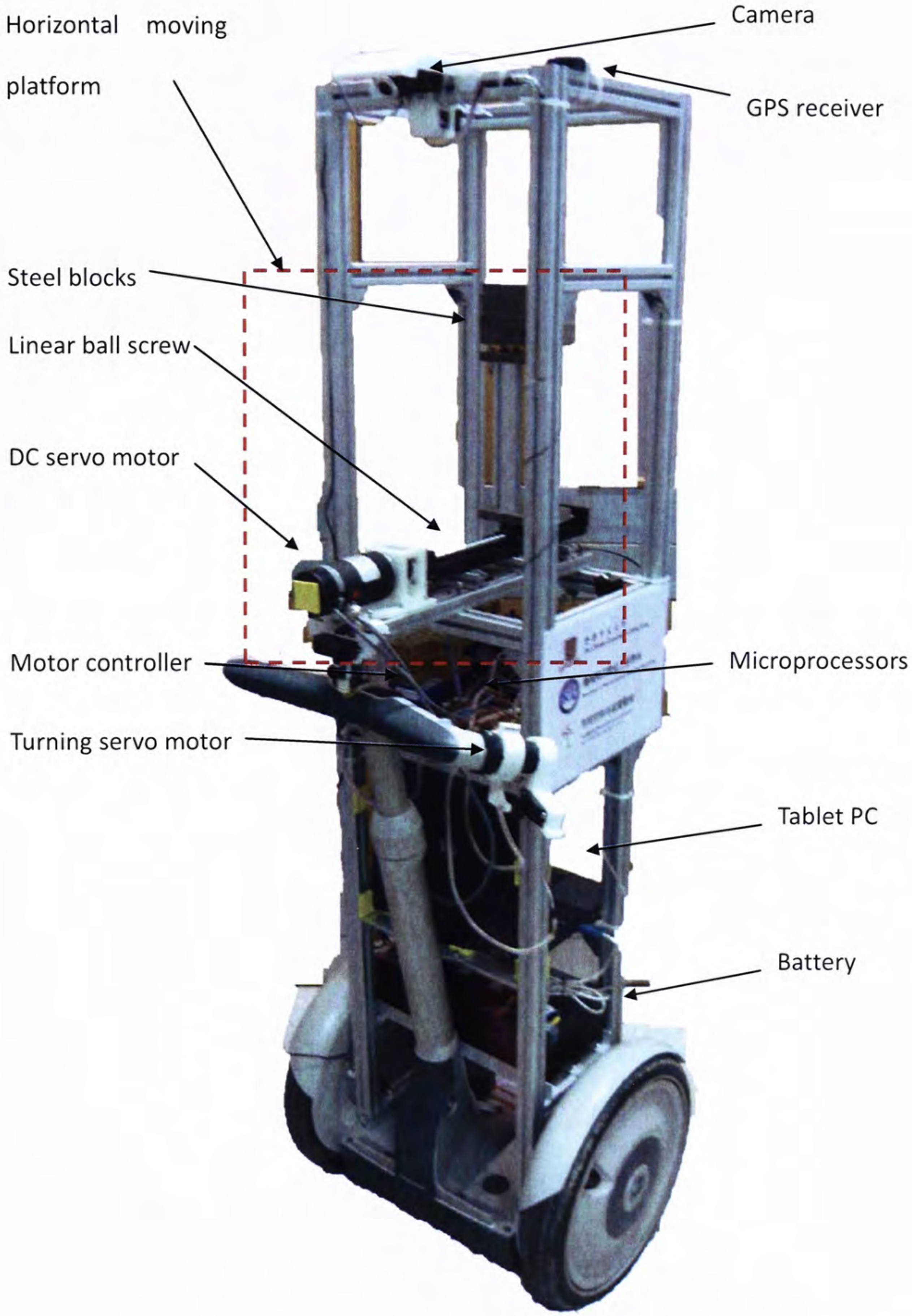

Figure 2.3. Position of components on the Segway Rider

2.3. Dynamic model of Segway system

The dynamic model of two-wheeled mobile system is well studied by other researches [12]. This type of system has three degree of freedom. It can be described by six state space variables.

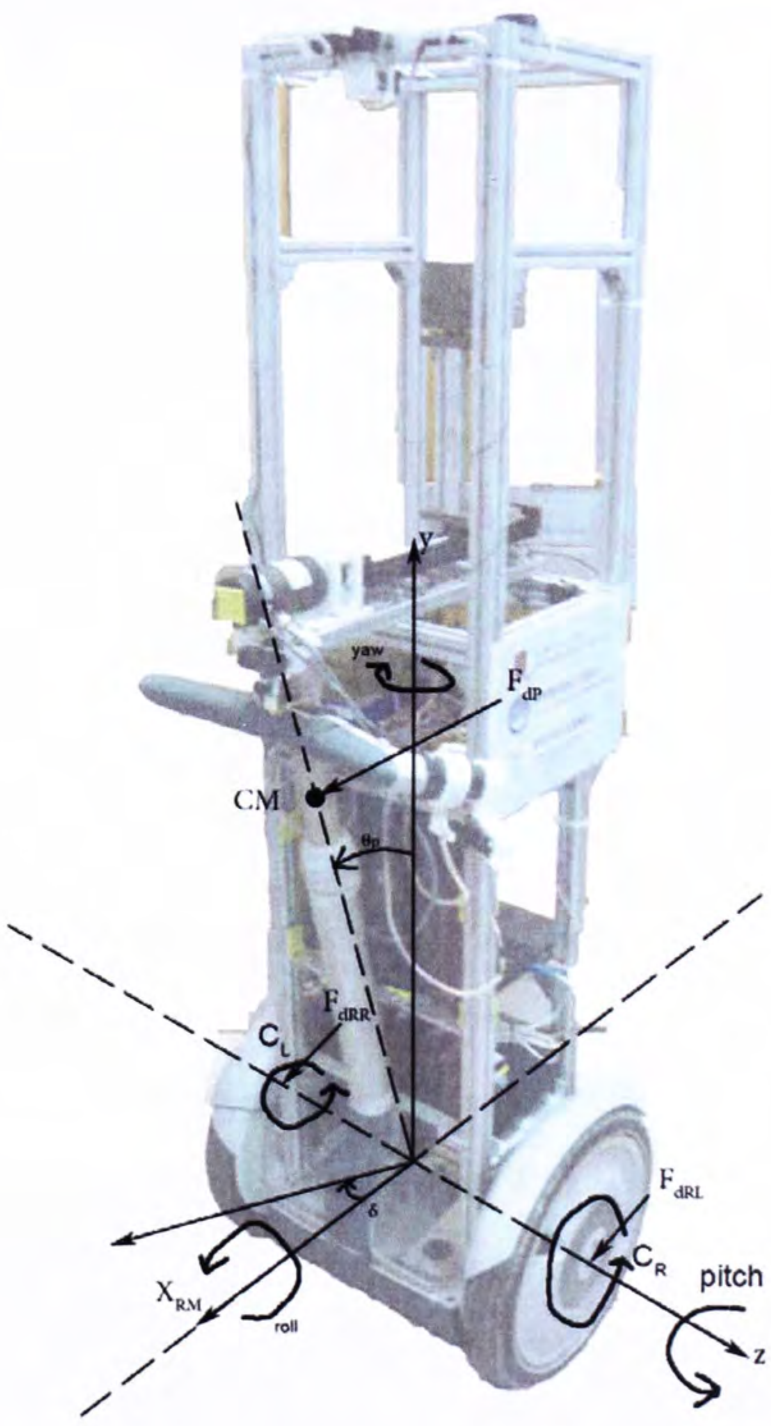

Figure 2.4. Definition of state space variables

The following six state space variables are chosen:

- The rotation around z-axis (pitch), θ_P
- The angular velocity of rotation around z-axis, $\dot{\theta}_P$
- The rotation around y-axis (yaw), δ_P
- The angular velocity of rotation around y-axis, $\dot{\delta}_P$
- Linear position of the chassis, x_{RM}
- Linear speed of the chassis, $\dot{x}_{RM}$

From the free body diagram of the wheels (Figure 2.5), equations for left wheel are shown below:

$$\ddot{x}_{RL} M_{RL} = F_{dRL} - H_L + H_{TL}$$

$$\ddot{y}_{RL} M_{RL} = V_{TL} - M_{RL} g + V_L$$

$$\ddot{\theta}_{RL} J_{RL} = C_L + H_{TL} R$$

$$\ddot{x}_{RR} M_{RR} = F_{dRR} - H_R + H_{TR}$$

$$\ddot{y}_{RR} M_{RR} = V_{TR} - M_{RR} g + V_R$$

$$\ddot{\theta}_{RR} J_{RR} = C_R + H_{TR} R$$

where

M_{RL}, M_{RL} are mass of rotating masses connecting to the wheels

$\ddot{x}_{RL}, \ddot{x}_{RR}$ are horizontal acceleaction of the wheels

$\ddot{y}_{RL}, \ddot{y}_{RR}$ are vertical acceleaction of the wheels

F_{dRL}, F_{dRR} are forces applied to center of gravity of the wheels

$\ddot{\theta}_{RL}, \ddot{\theta}_{RR}$ are angular acceleaction of the wheels

J_{RL}, J_{RR} are moment of inertia of the rotating masses with respect to $z - $axis

R is radius of the wheels (assume both wheels are equivalent)

$H_L, H_R, H_{TL}, H_{TR}, V_{TL}, V_{TR}, V_L, V_R$ are reaction forces between different bodies

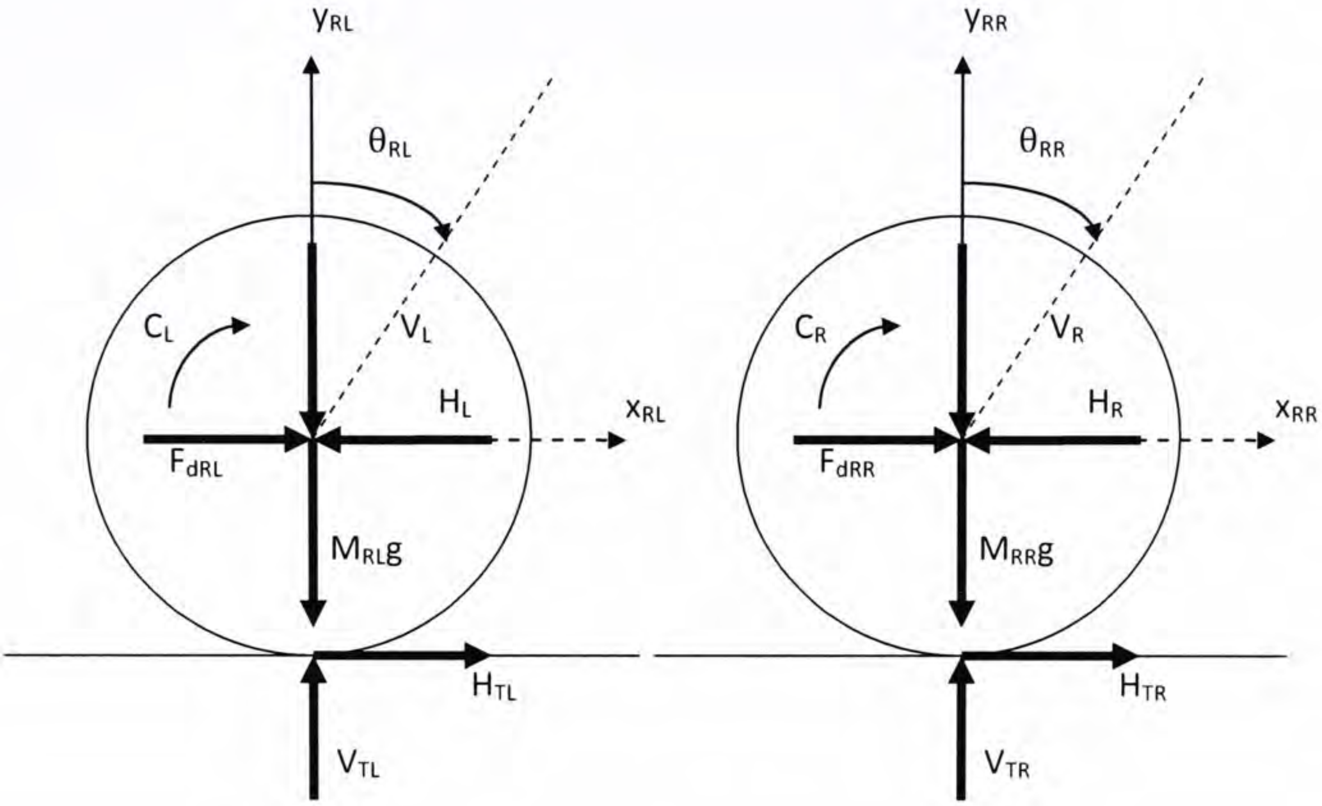

Figure 2.5. Free body diagram of wheels

From the free body diagram of the system (Figure 2.6), equations for the system are shown below:

$$\ddot{x}_P M_P = F_{dP} - H_R + H_L$$

$$\ddot{y}_P M_P = V_R + V_L - M_P g - F_{C\theta}$$

$$\ddot{\theta}_P J_{P\theta} = (V_L + V_R) L \sin\theta_P - (H_L + H_R) L \cos\theta_P - (C_L + C_R)$$

$$\ddot{\delta} J_{P\delta} = (H_L + H_R)\,\frac{D}{2}$$

where

M_P is mass of chassis

F_{dP} is forces applied to center of gravity of the system

$J_{P\theta}$ is moment of inertia of the chassis with respect to z − axis

$J_{P\delta}$ is moment of inertia of the chassis with respect to y − axis

L is distance between center of gravity of the chassis and $z-$ axis

D is distance between two wheels

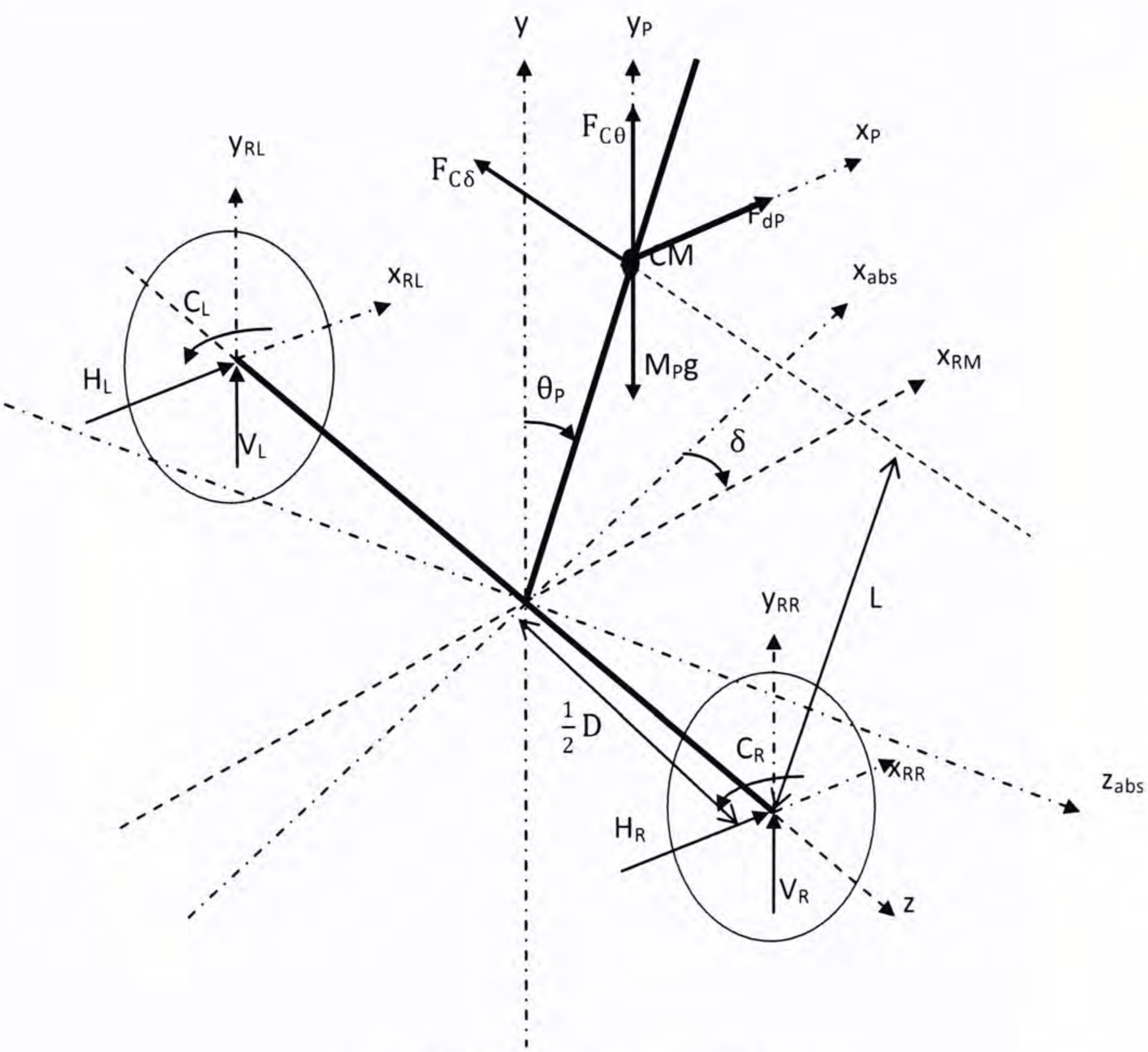

Figure 2.6. Free body diagram of system

2.4. Actuating components

The framework structure of the Segway Rider is built by 30mm X 30mm aluminum profile. The dimension of the Rider is 28.5 cm x 33 cm x 150cm. The Rider is mounted on the Segway PT by screws between the bottom of the framework and the Segway PT. The framework is shown on Figure 2.7.

Figure 2.7. Empty profile framework mounted on Segway PT

The horizontal moving platform shown in Figure 2.8 is designed for implementing the core control part of the Rider, which is moving the mass placed on the platform in order to change the center of mass of the entire system. It consists of a NSK 300mm long ball screw, a DC servo motor, and 7 steel blocks which are 5.6kg in total (0.8kg each).

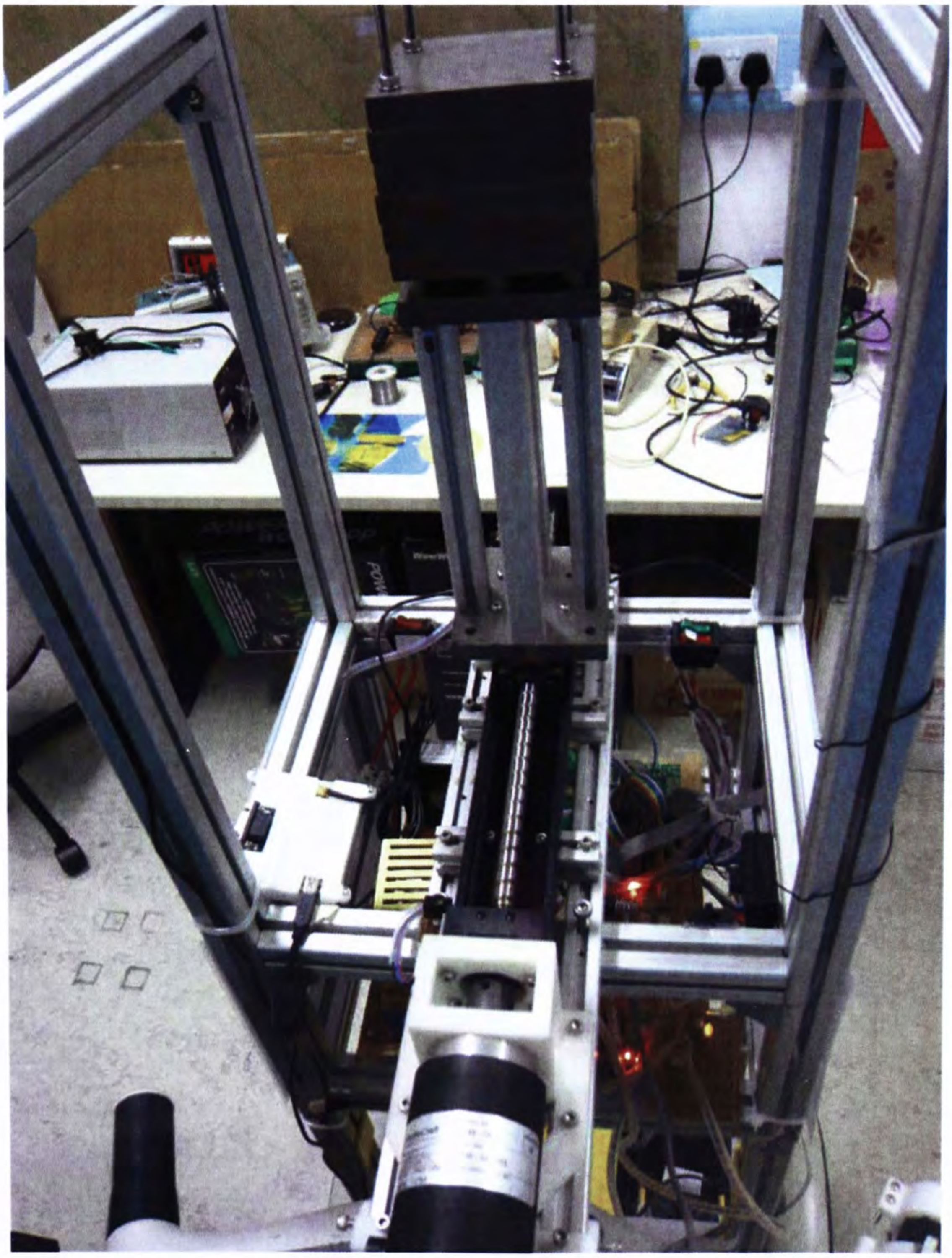

Figure 2.8(a). Horizontal moving platform overview

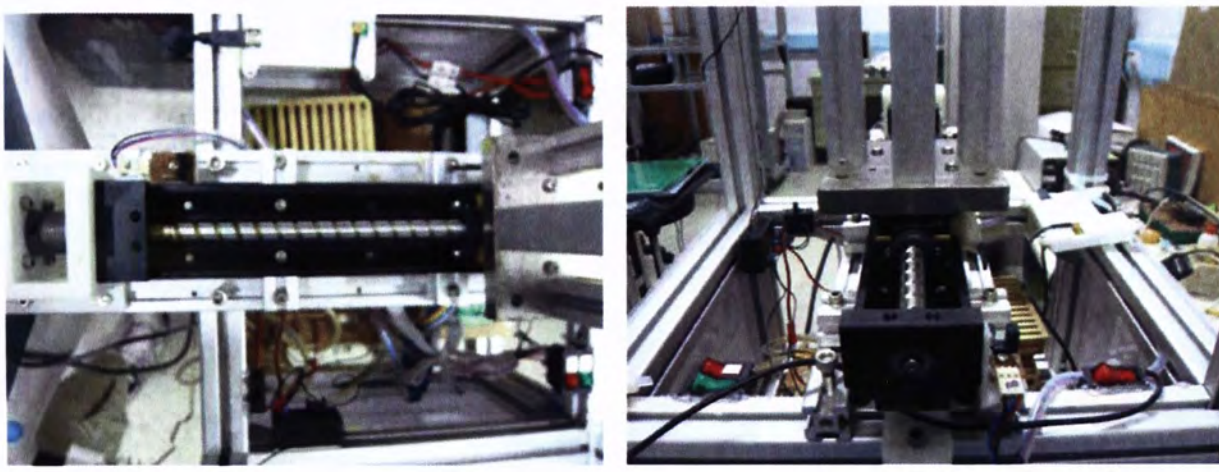

Figure 2.8(b). Linear ball screw

Figure 2.8(c). DC Servo motor

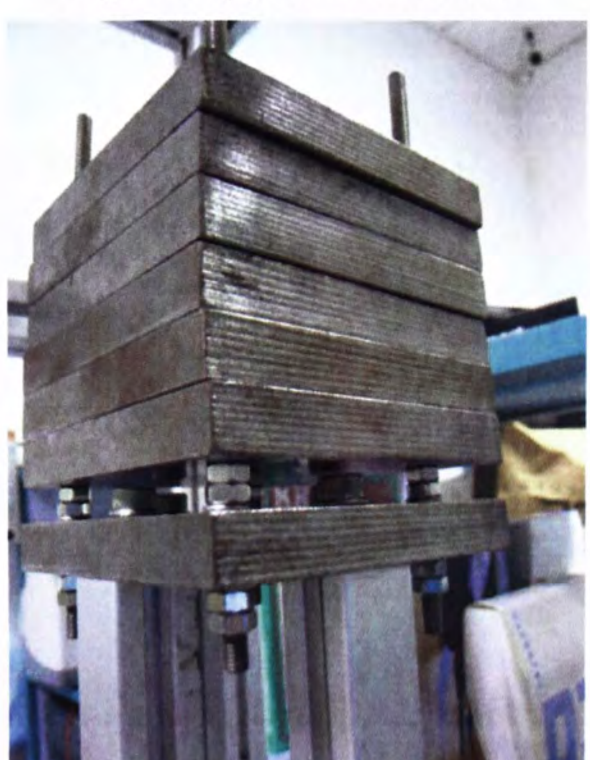

Figure 2.8(d). Steel blocks

For safety purpose, two infra-red limit switches shown in Figure 2.9 are installed and connected to motor controller in order to avoid burn-out of motor and other components. If anyone of the switches is triggered, further motion on that direction is prohibited. Moreover, one of the limit switches serves as the zero reference sensor for coordinate reference setting of the motor controller.

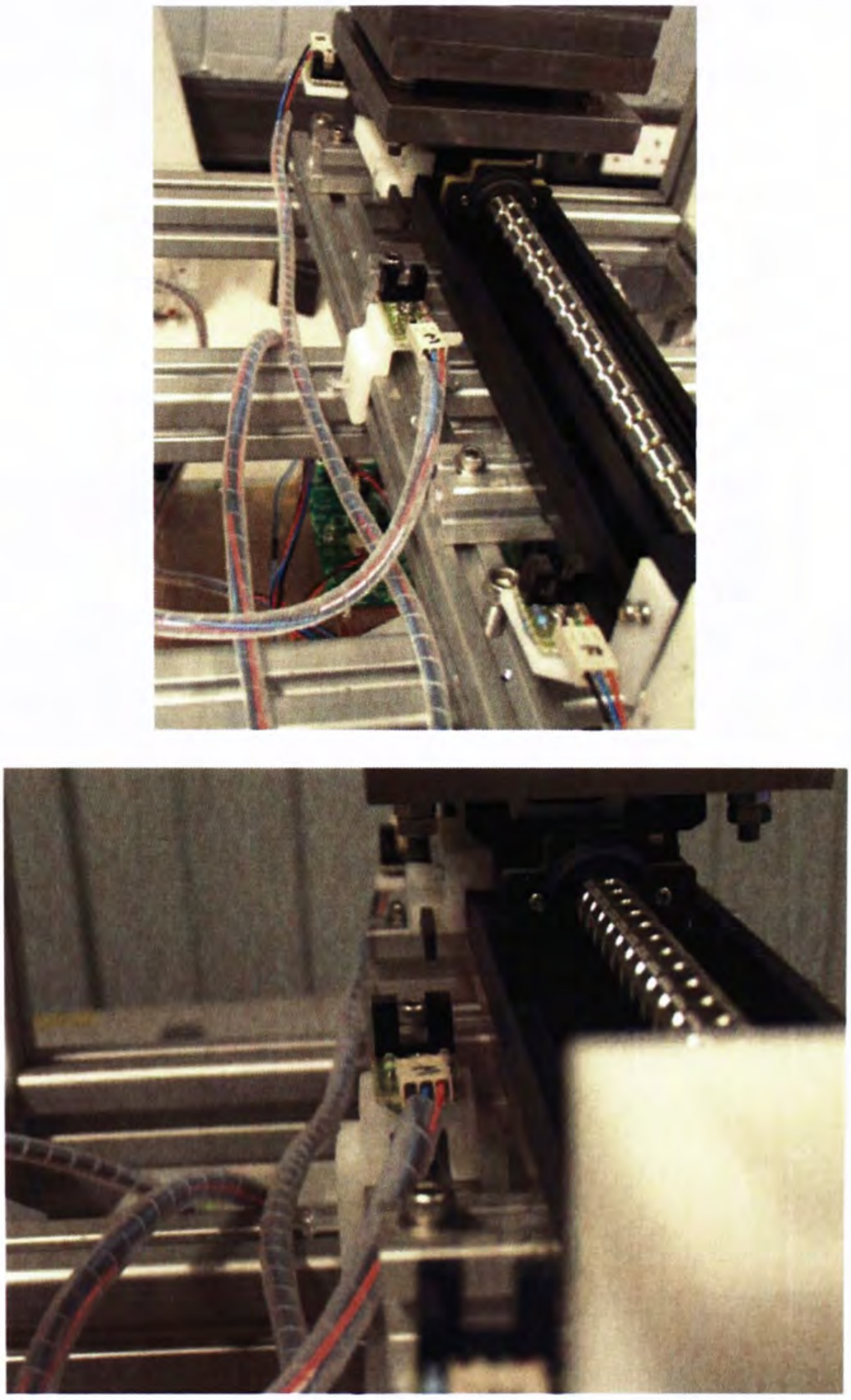

Figure 2.9. (a, b) Limit switches attached on linear ball screw

The servo motor used on horizontal moving platform is controlled by a motor controller. The motor controller is shown in Figure 2.10. The model of the controller is Accelnet ACM-055-18 by Copley Controls Corp. It can be controlled by CAN, USB or serial interface. Serial interface is chosen for our Rider. This motor controller is capable for high voltage and high current such that it is ready for future development of higher loading of weights. Heat sink and cooling fan are installed for cooling down the heat generated during operation. The detailed specification of the motor controller is listed as follows:

Operating voltage: 20-55 V

Continuous current: 6 A

Continuous Output Power 330 W

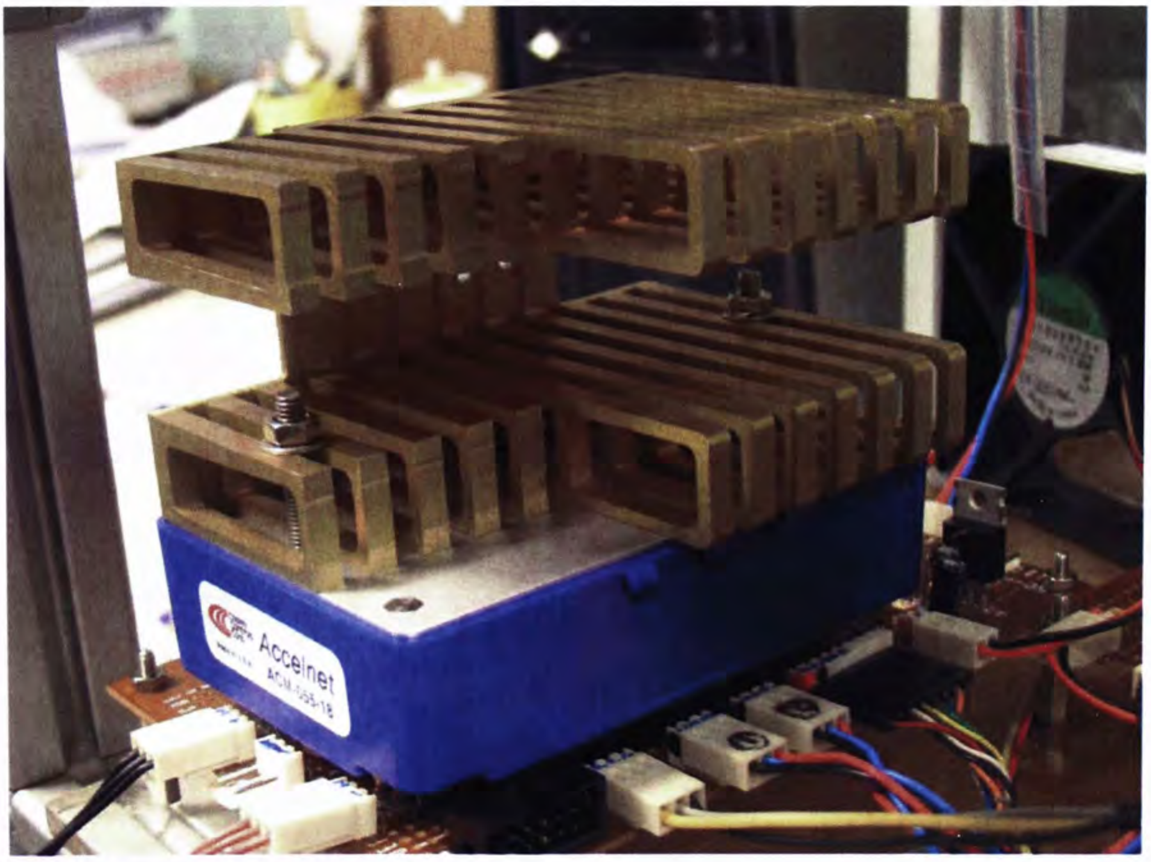

Figure 2.10. Accelnet ACM-055-18 Motor controller with heat sink

Turning motion of Segway PT i-series is done by rotating a switch by hand which is shown on Figure 2.11. In order to implement the turning function on the Rider, a model-grade DC Brushless servo motor is installed on the Rider by a mounting. The specification of the motor is listed as below:

Model: Futaba BLS 352

Speed: 0.15 sec/60°

Torque: 18 kg • cm

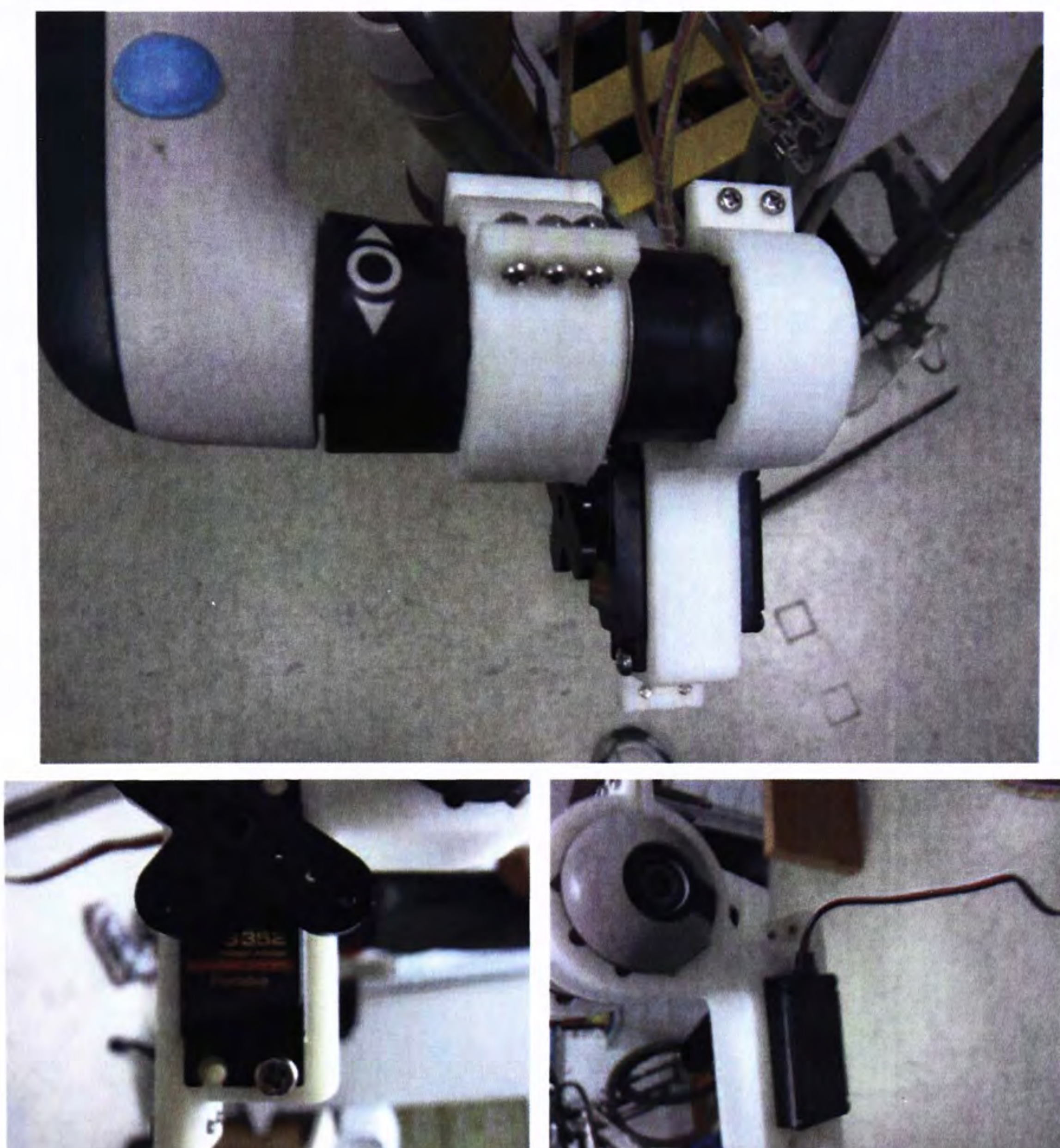

Figure 2.11. Turning mechanism with DC motor and mounting

2.4. Electronic and sensing components

Two microprocessors are installed on the Rider to gather sensors information and control the position of the mass on the linear ball screw. Atmel ATMega8535L is used for handling sensors, while Atmel ATMega128L is used for motor control and handling information flow between laptop, microprocessors and motor controller.

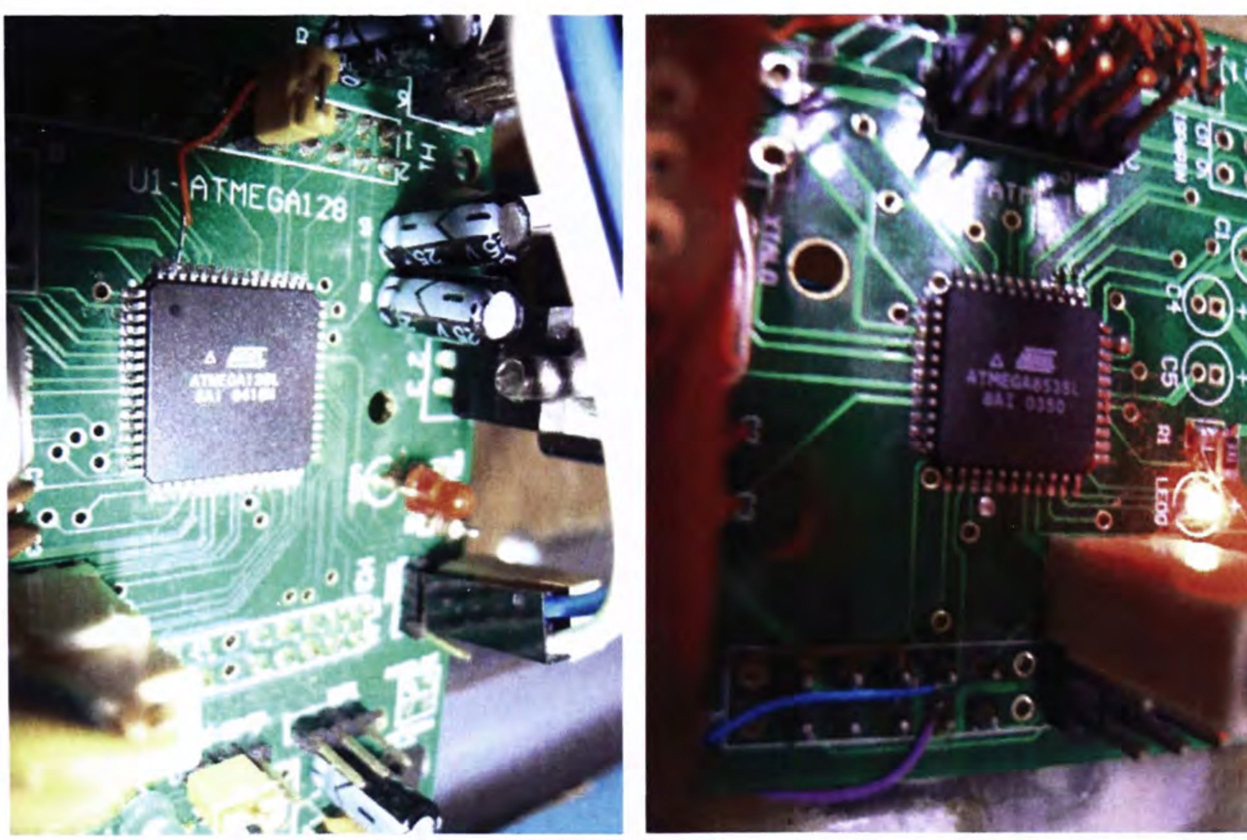

Figure 2.12. ATMega128L (left); ATMega8535L (right)

The Rider is equipped with a laptop computer for real-time video processing, GPS data gathering and real-time decision making. The detailed specification of the laptop is listed as follows:

Model: HP HSTNN-C02C Tablet PC

CPU: T2300 @1.66GHz

Ram: 1.5GB DDR Ram

Harddisk: 60GB

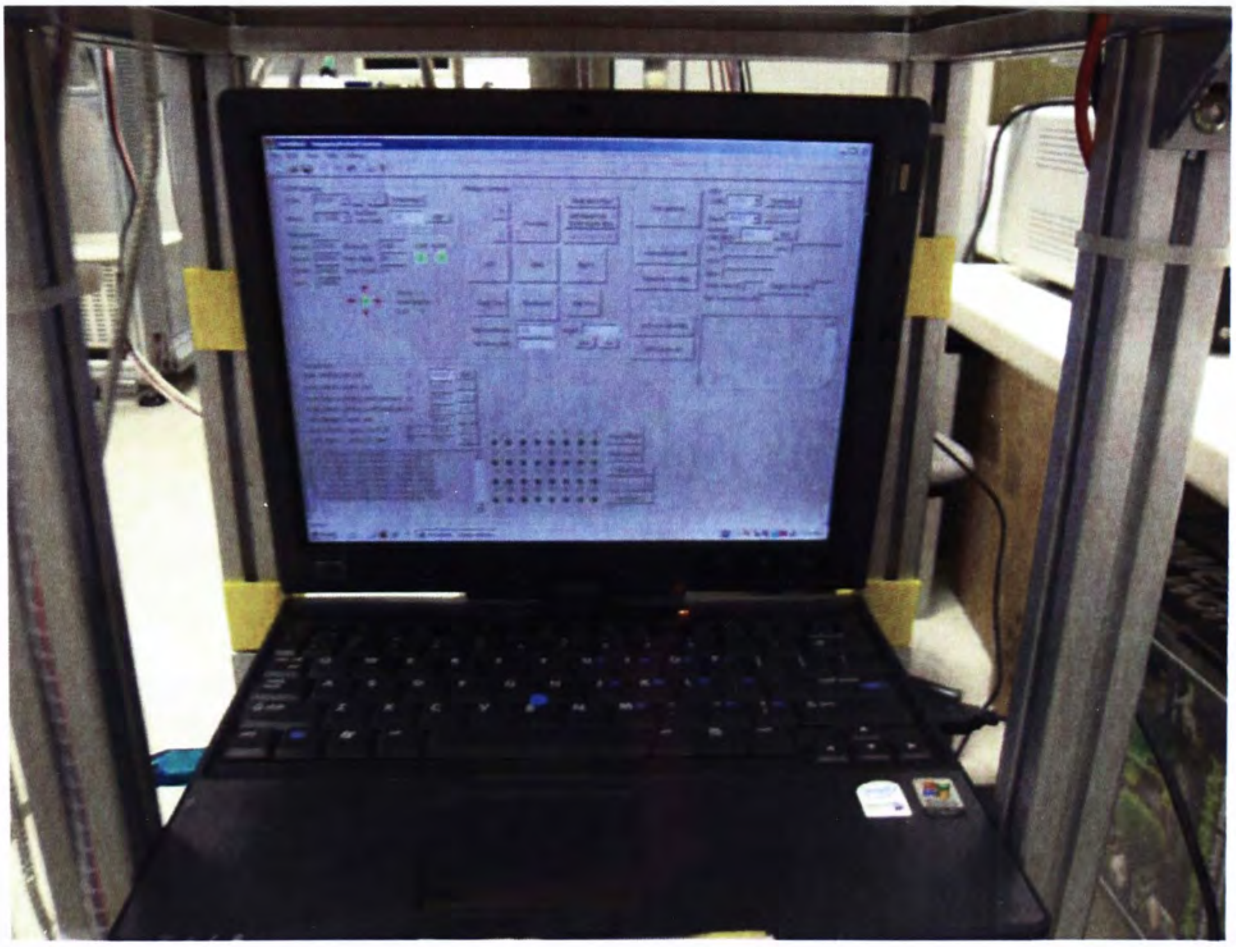

Figure 2.13. HP laptop operating on the Rider

A Logitech QuickCam Pro 9000 webcam is mounted on the Rider for real-time video capturing. For reasonable use of processing power, the video resolution used is 24-bit 320x240 pixels. However, the camera supports following standard:

Resolution: Up to 2 million pixels (1920 * 1200)

Frame rate: 30 frames per second

Figure 2.14. Logitech QuickCam Pro 9000 mounted on the Rider

The GPS receiver equipped on the Rider uses Antaris 4 chips developed by Atmel and u-Blux shown in Figure 2.15. It gathers information from both American GPS satellites and European Galileo system. For this reason, it can achieve higher horizontal accuracy than traditional GPS receiver. Its average horizontal error is 1.5m.

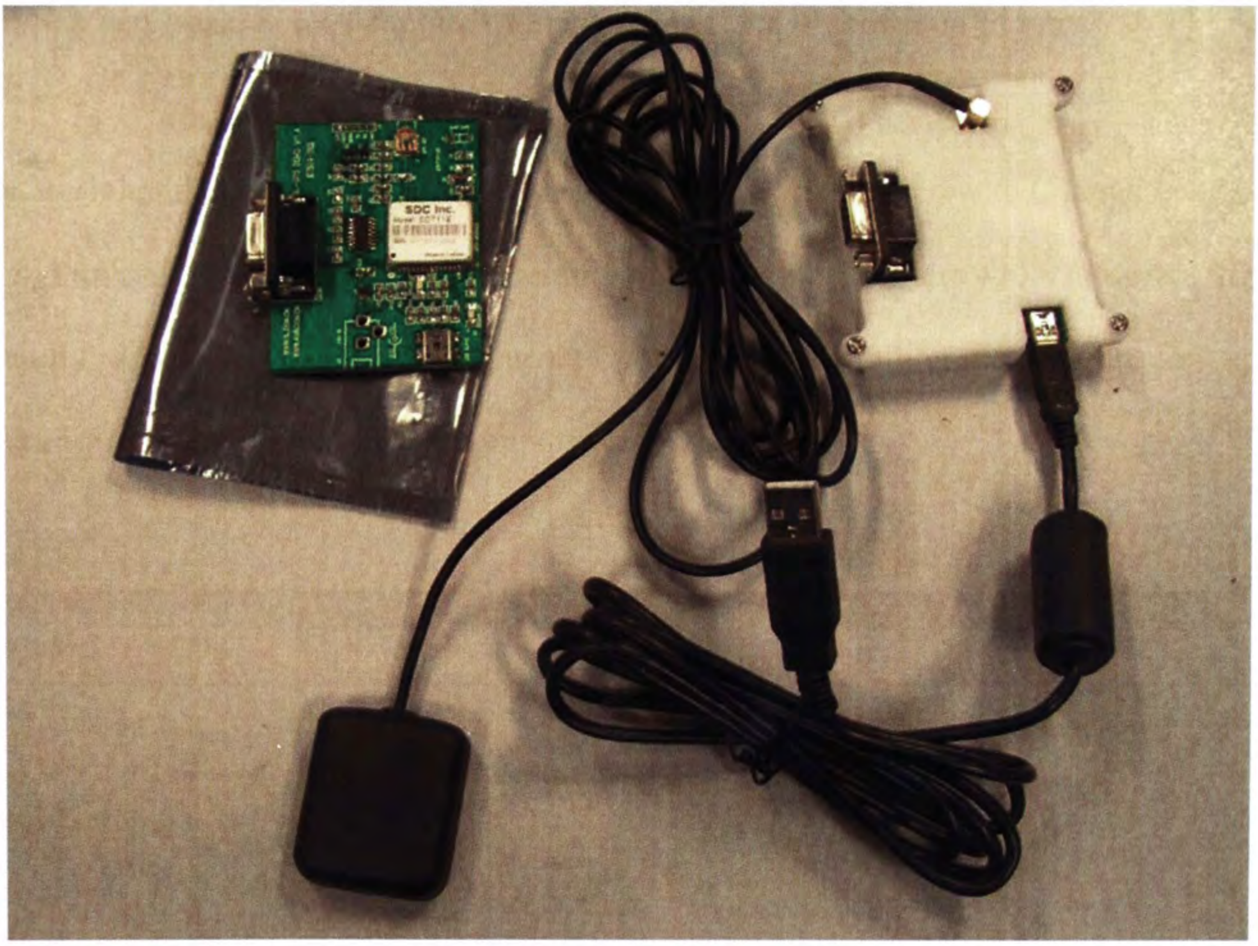

Figure 2.15. GPS receiver used on the Segway Rider

2.5. Software development of Segway Rider

Software development is separated into two parts. Software used on the laptop computer is Windows XP based and developed by Visual Studio 2005 which is coded by C language. It is mainly used for gathering GPS information and sensors information returned from microprocessors, processing real-time video and making motion decisions. Screen capture of the user interface is shown on Figure 2.16.

On the other hand, software used inside microprocessors is coded with C language. The program can be flashed into ROM memory of the microprocessors while modified. It is mainly responsible for handling sensors information, controlling motor controller and returning information to laptop computer.

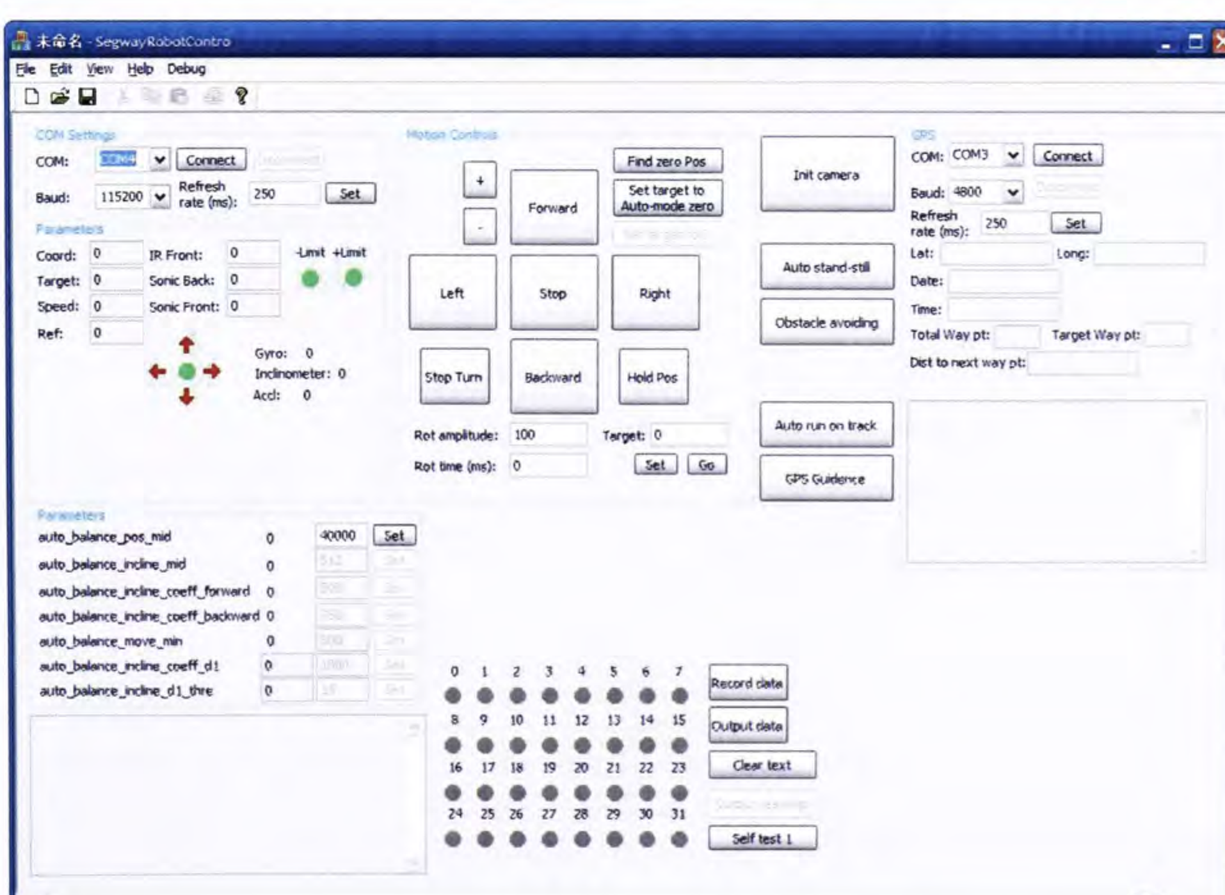

Figure 2.16. User interface to control Segway Rider

In order to link up laptop, microprocessors and external sensing devices, serial port through RS-232 interface is used for communications between laptop-microprocessor and microprocessor-motor controller connections. Also, I^2C protocol is used for inter-microprocessor and microprocessor-sensors communications. The flowchart of software data flow is shown on Figure 2.17.

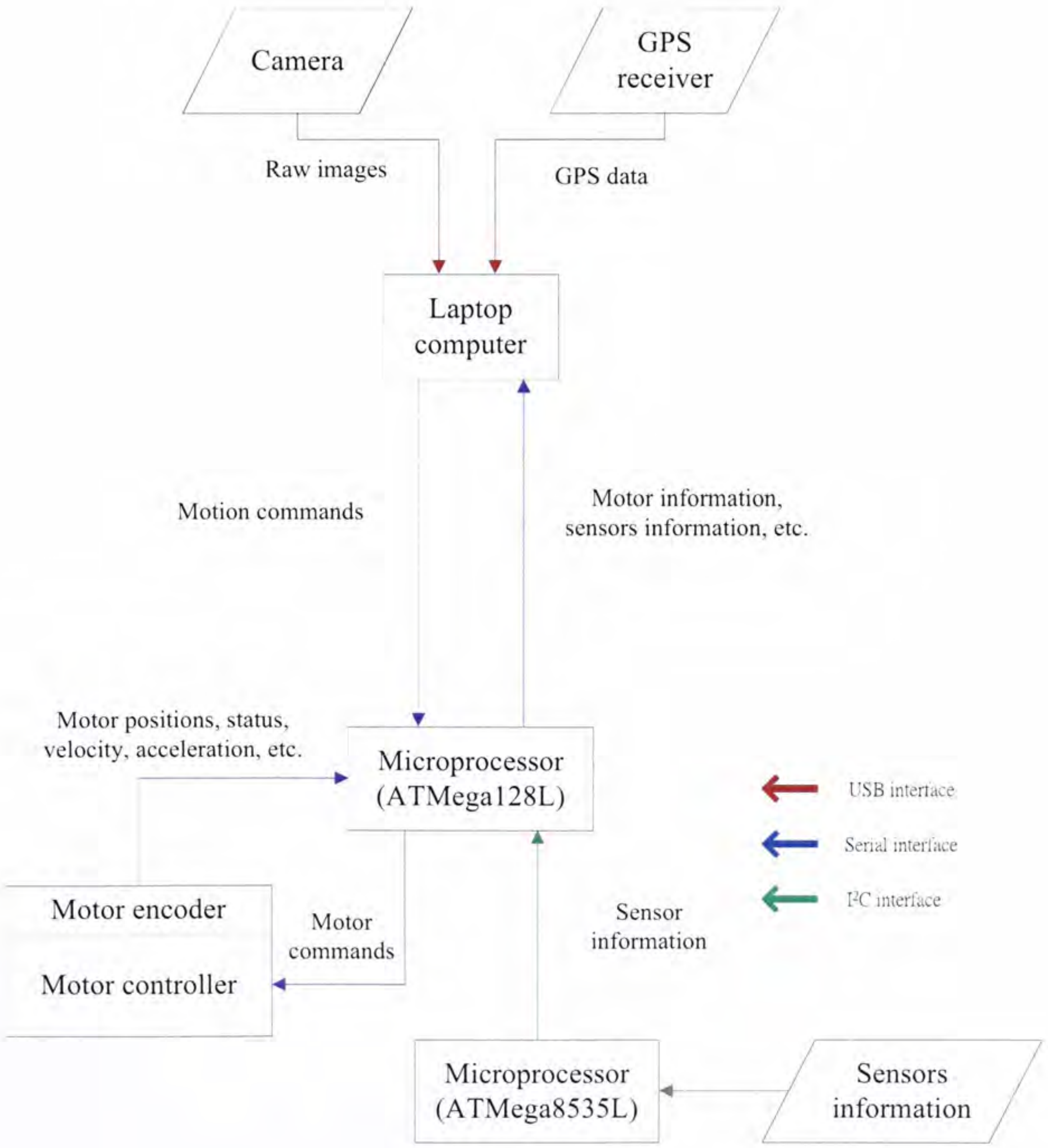

Figure 2.17. Software flowchart of the Segway Rider

Since the software of the Segway Rider is developed by Visual C++ 2005, many third parties API can be imported to our software. For example, OpenCV is used for real-time computer vision processing for the Rider. In the future, more functions can be added to the software by implementing the functions using different third parties API. Internet access is a possible feature that can be implemented on the Rider.

2.6. Chapter summary

The Segway Rider aimed at constructing a mobility robotics platform that is fitted for different mobility platform researches. With the advantage of small footprint, zero-turning radius and high payload, the Rider is suitable for conducting researches for both indoor and outdoor environment. The Rider is designed with board extension ability. Lots of space is available for installing more sensors and equipments. This Chapter introduces the hardware components and their specifications, and also the usage of the components is described in detail. The software of the Rider is also introduced with flowchart of data flow.

Chapter 3

The grand challenge

3.1. Objective

The objective of this project is to construct a robotics mobility platform that can be used as research platform for either outdoor or indoor environment. As the first research based on the Segway Rider, we try to implement outdoor auto-navigation for our project.

To fulfill the above requirement, we implement functions that allow the Rider to run on the running track of the CUHK University Gymnasium shown in Figure 3.1 automatically. The Segway is required to complete the task with a single "start" command. It needs to keep itself on the same lane within a 400 meters complete loop. It is also required to operate its task in fully automatically behavior.

Figure 3.1. CUHK University Gymnasium running track

3.2. Experiment

The grand challenge is successfully finished on 27 April, 2009 at the CUHK University Gymnasium running track. Total time needed for completing a loop on the track is 5 minutes and 25 seconds. We use only Computer Vision technique to keep the Segway Rider on track while GPS data is used for global positioning.

The experiment will be discussed as follows. First part will describe the computer vision methods that are implemented on the Rider. Second part will show the use of GPS data for global positioning. The last part will show what other functions are installed on the Segway Rider.

Figure 3.2. Snapshot of the video for the Grand challenge

3.3. Running lane tracking by computer vision

The concept of the algorithm for keeping the Rider on same lane is rotating the Rider towards opposition while the angle between the direction of the Rider and the tangent of the lane is too large. For this reason, the Rider is running on sig-sag behavior to keep itself towards the center of the lane. Figure 3.3 shows the calculation record of the algorithm.

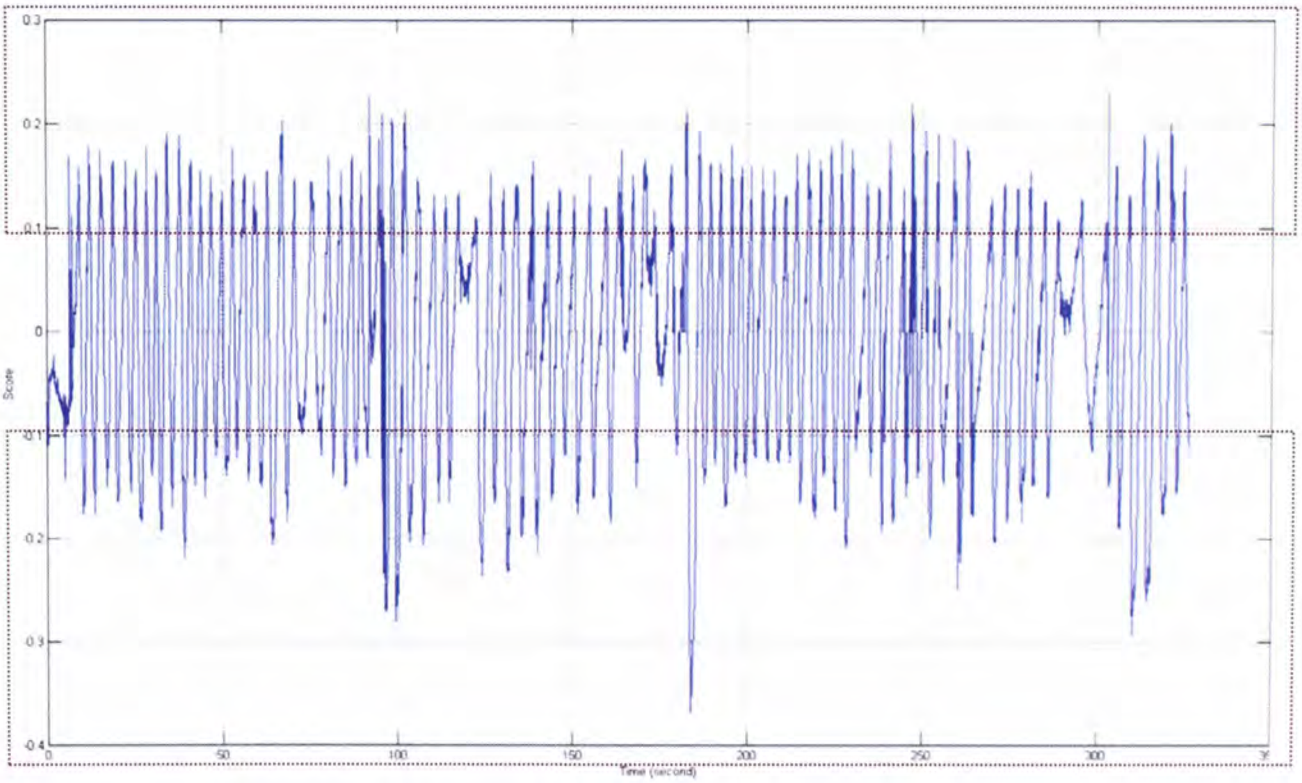

Figure 3.3. Plot of calculation result

The threshold set for the Rider response is 0.1 score (red area in figure 3.3). When the calculated score is larger than, it means the Rider needs to turn right, and vice versa. Sig-sag pattern is observed in the graph obviously.

In order to keep the Rider to move on the same lane within a 400 meters loop, several steps of computer vision processes are needed for the Rider to determine its motion needed to respond.

The frame rate we used for processing is 10 frames / second, which is enough for mechanical response of the Rider. The image resolution used is 320x240 pixels. For each frame of real-time capturing video, five main steps are implemented.

3.3.1. Color space conversion

For most webcam, image captured is represented in RGB color-space. As our task needs video processing on running track, which is an environment mainly consisting red color for the background color and white color for the lines of lanes. We choose YCrCb color space for processing. Only Cr channel is picked for further process. The Cr channel in YCrCb color space represents red-difference chrome components. The equation for RGB to YCrCb conversion is shown below:

$$Y = 0.299 \times R + 0.587 \times G + 0.114 \times B$$

$$Cr = (R - Y) \times 0.713 + 0.5$$

$$Cb = (B - Y) \times 0.564 + 0.5$$

where R, G, B represents red, green and blue component for RGB image respectively;

Y, Cr, Cb represents luminance, red-difference and blue-difference component for YCrCb image respectively

3.3.2. Apply binary threshold

After color space conversion, image is applied with binary threshold cutoff. The threshold value is a pre-set which is capable for most environment conditions, such as wet floor, dirty textures on the track and different direction of sunlight. These conditions affect the color of the input image especially for image in RGB color space, which is not so obvious on YCrCb color space. By applying the binary threshold cutoff, the patterns of lanes become clear on the image shown as Figure 3.4.

There are two examples for comparing the image results after applying binary threshold cutoff for using RGB color space image and YCrCb color space image. Example 1 shows the Rider is moving on lane with small curvature, see Figure 3.5. The floor is in dry condition with some irregular texture on the track. The result images show that binary cut off on RGB work well for the white lines on track. However, it also generates patterns for unwanted features on the track. For binary cut off using YCrCb image, the process successfully filter out unwanted patterns with acceptable performance to show the white lines.

Example 2 shows an extreme case when the track is wet, see Figure 3.6. The incoming image is obviously darkened. The result of image in RGB is seriously affected. The line pattern is lost in upper half of the image. In contrast, image using YCrCb still keep acceptable performance on showing the lines.

It is important for us to keep the mechanism working well on different environment because the task is running in outdoor, public and open area. Lots of uncertainty may happen during our experiment.

Figure 3.4. Intermediate images after binary cutoff process; Original image in RGB (top); image in Cr channel (middle); image after binary cutoff (bottom)

Comparison example 1: Lanes with small curvature

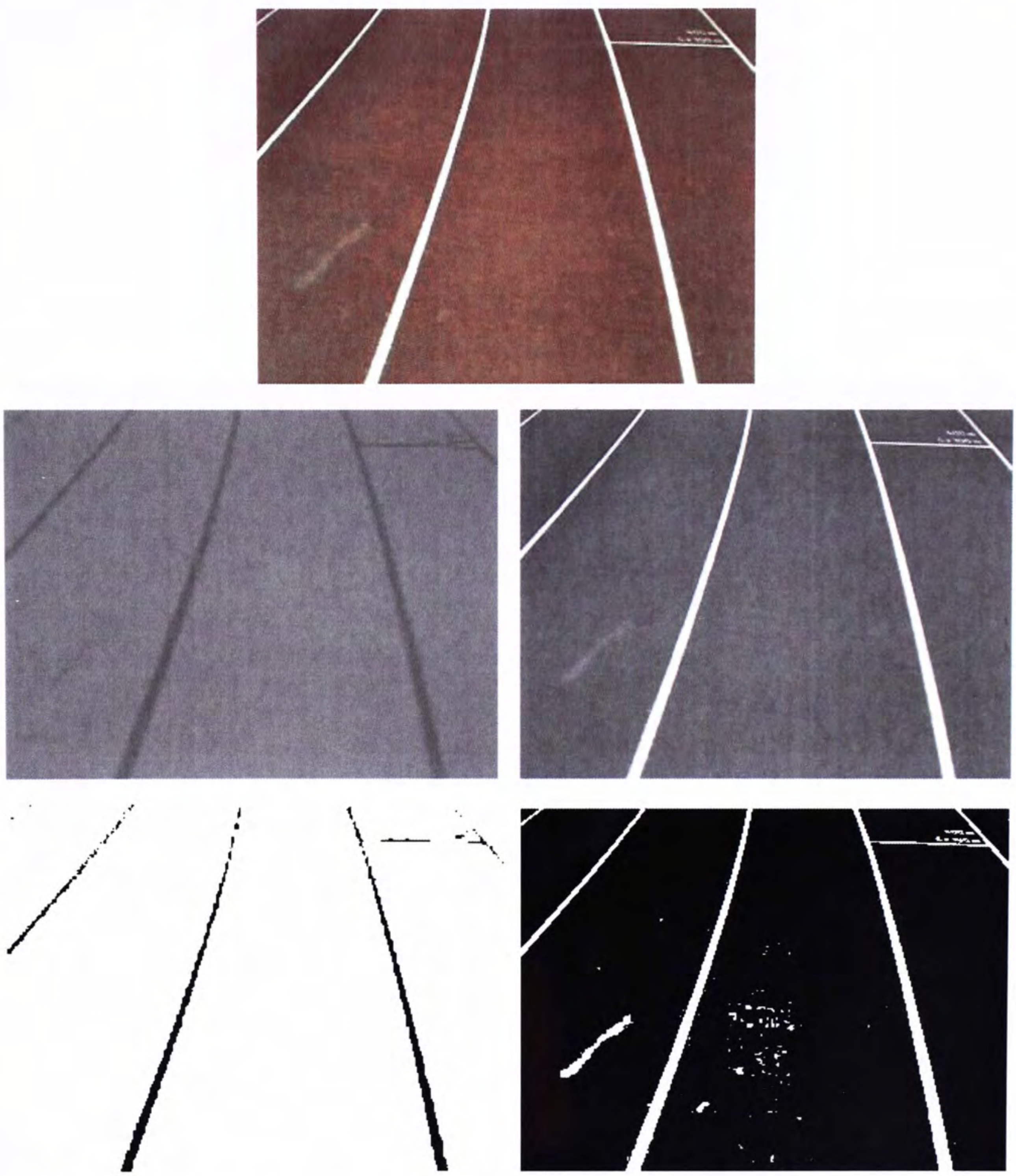

Figure 3.5. Comparison on processing result using YCrCb and RGB color space

Top: Original image;

Middle: Cr channel (left); Grayscale image (right)

Bottom: Cr with binary threshold (left); Grayscale with binary threshold (right)

Comparison example 2: Wet running track floor

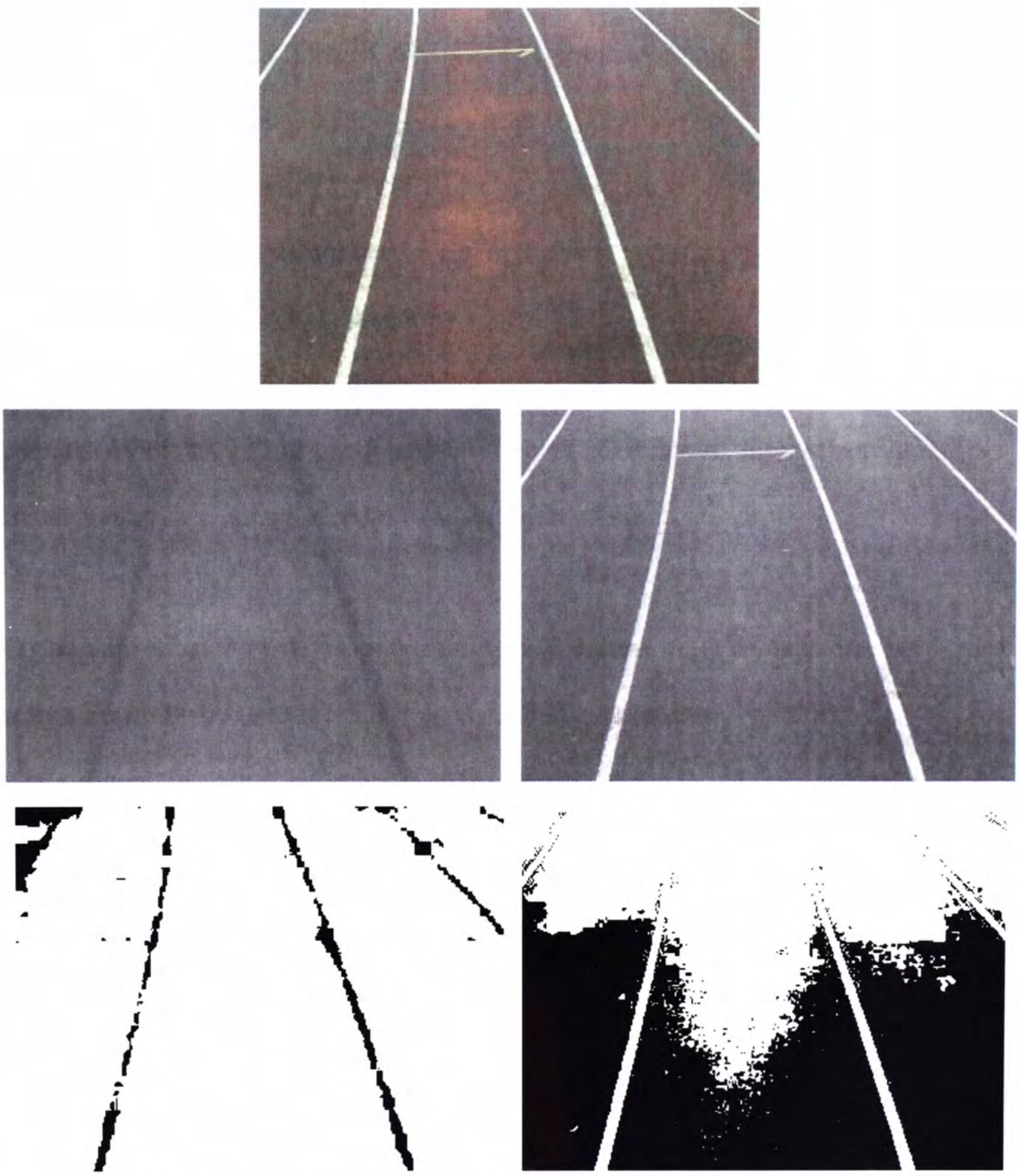

Figure 3.6. Comparison on processing result using YCrCb and RGB color space

Top: Original image;

Middle: Cr channel (left); Grayscale image (right)

Bottom: Cr with binary threshold (left); Grayscale with binary threshold (right)

3.3.3. Edge detection

While first two steps are finished, we have a clear binary image with line pattern on it. To determine whether the Rider is offset from the center of lane or not, left and right line information is needed for the Rider to make decision. By using Canny edge detector, point information about edges in image is recovered.

From the examples shown in Figure 3.7, line pattern can be recovered even the image generated by binary cut off is not perfect. It gives flexibility to the mechanism that we can use more relax setting in order to adapt different environment.

For the convenience of developing in OpenCV, Canny edge detector is used in our project. Several types of edge detecting algorithm is tested in our platform. There is not much different in performance between the algorithms.

Three edge detectors are tested on our platform, which are Canny, Sobel and Laplacian. Two set of examples is shown on Figure 3.8. All results in same column are processed using same image with binary cut off.

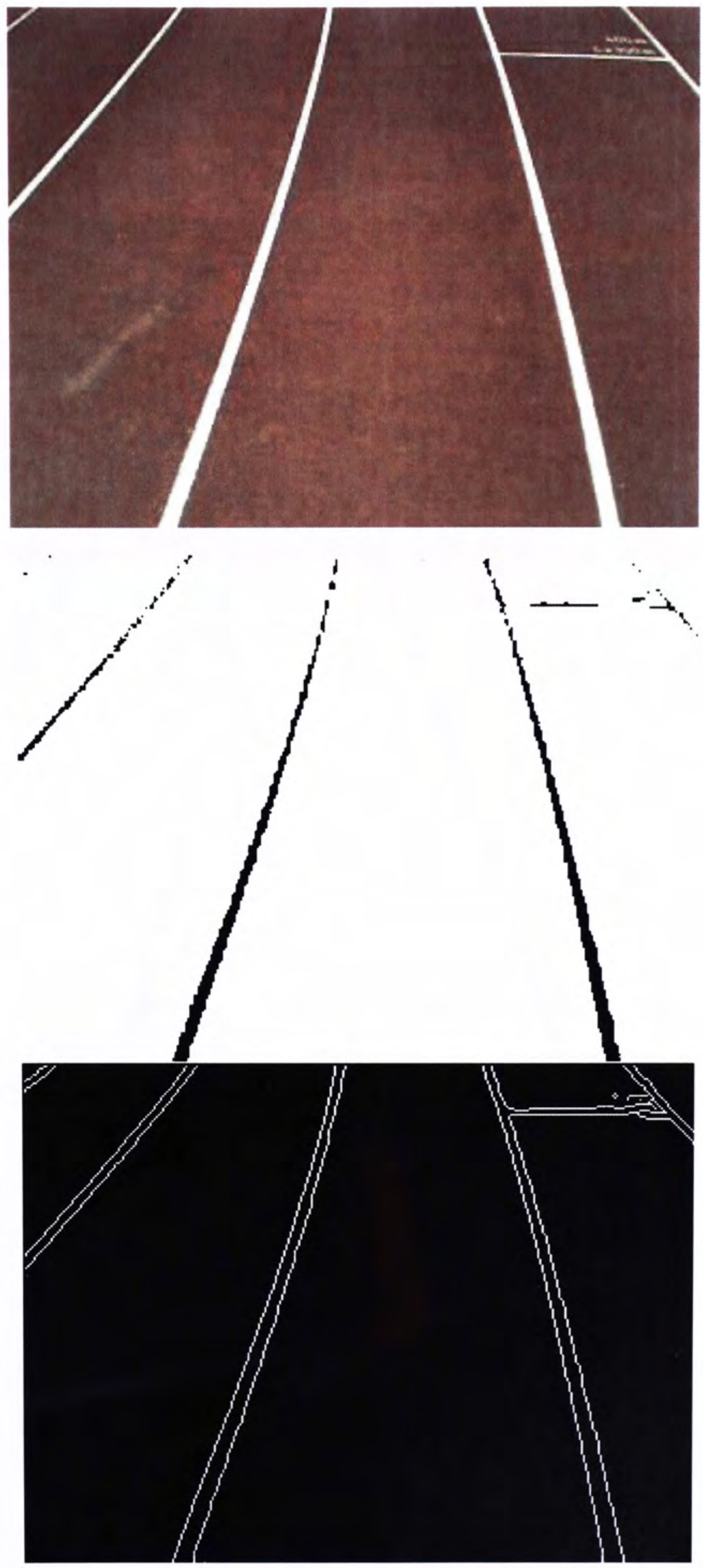

Figure 3.7(a). Intermediate images after Canny edge detection

Sample image in RGB (top); image after binary cutoff (middle); image with Canny edge (bottom)

Figure 3.7(b). Intermediate images after Canny edge detection

Sample image in RGB (top); image after binary cutoff (middle);

image with Canny edge (bottom)

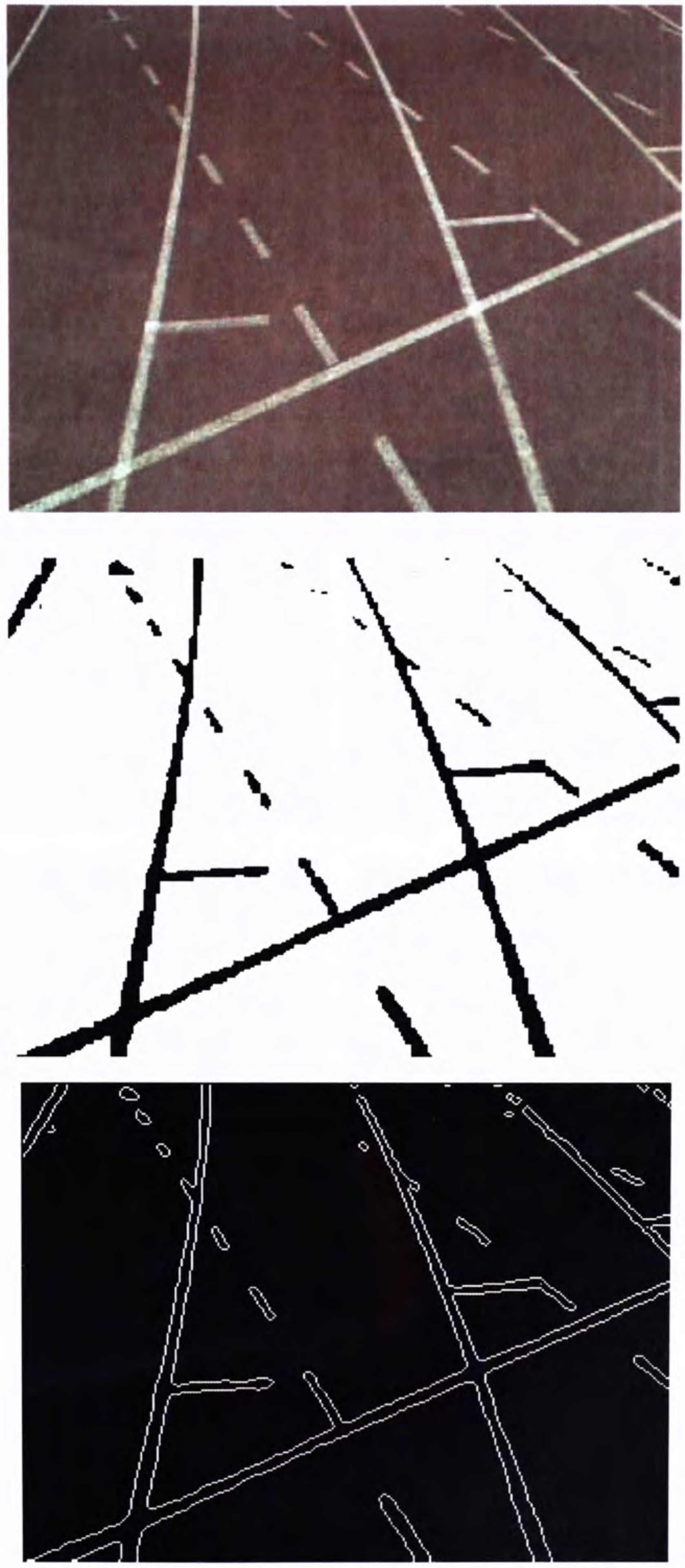

Figure 3.7(c). Intermediate images after Canny edge detection

Sample image in RGB (top); image after binary cutoff (middle);

image with Canny edge (bottom)

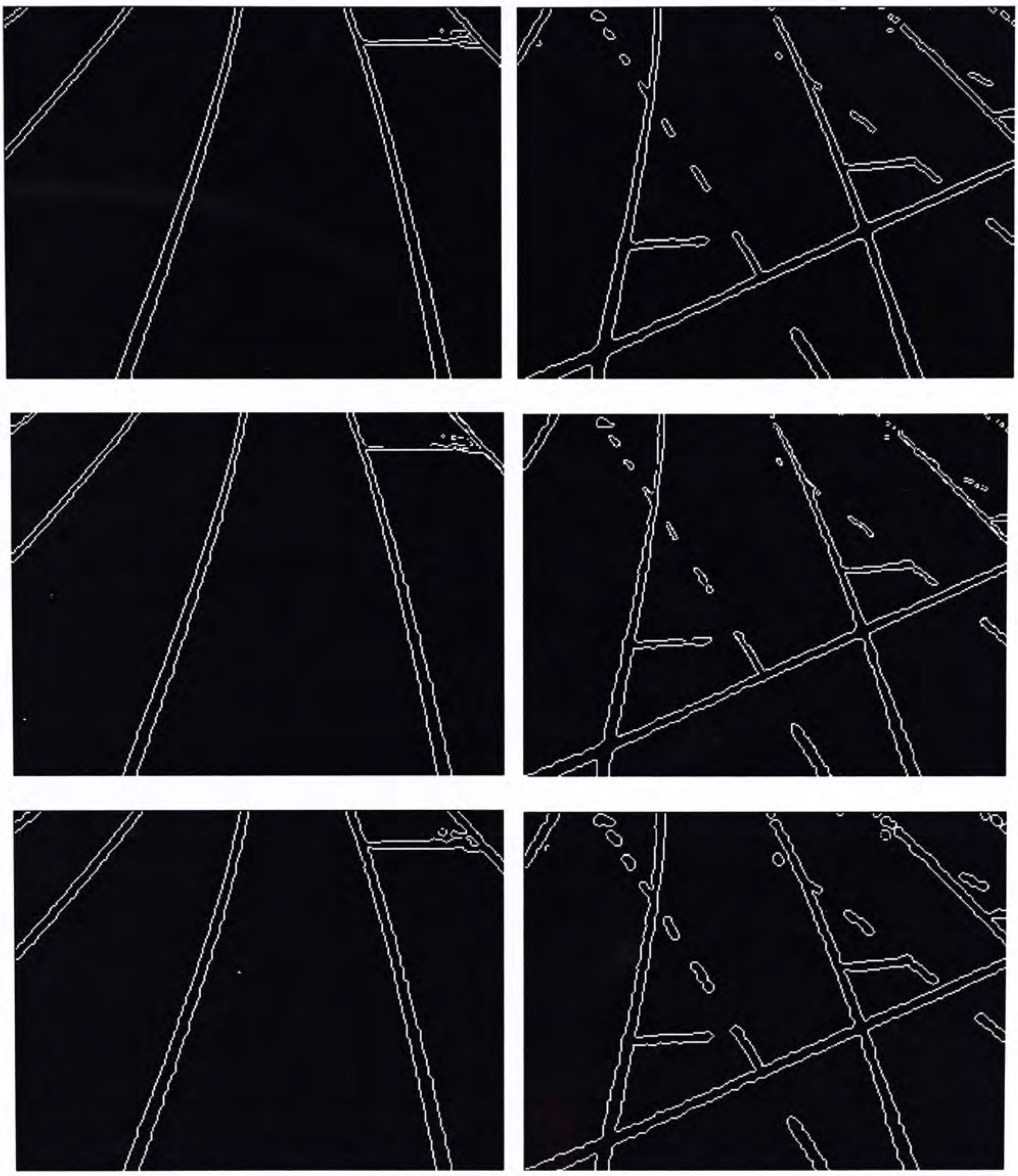

Figure 3.8. Comparison on different edge detection methods

Top: Canny; Middle: Sobel; Bottom: Laplacian

3.3.4. Hough transform

Generalized Hough transform was invented by Richard Duda and Peter Hart in 1972. It is commonly used for identifying lines in image processing. This method can identify imperfect line information from images caused by edge detector or small obstacles. Automated analysis can be done by voting in parameter space to find out line information.

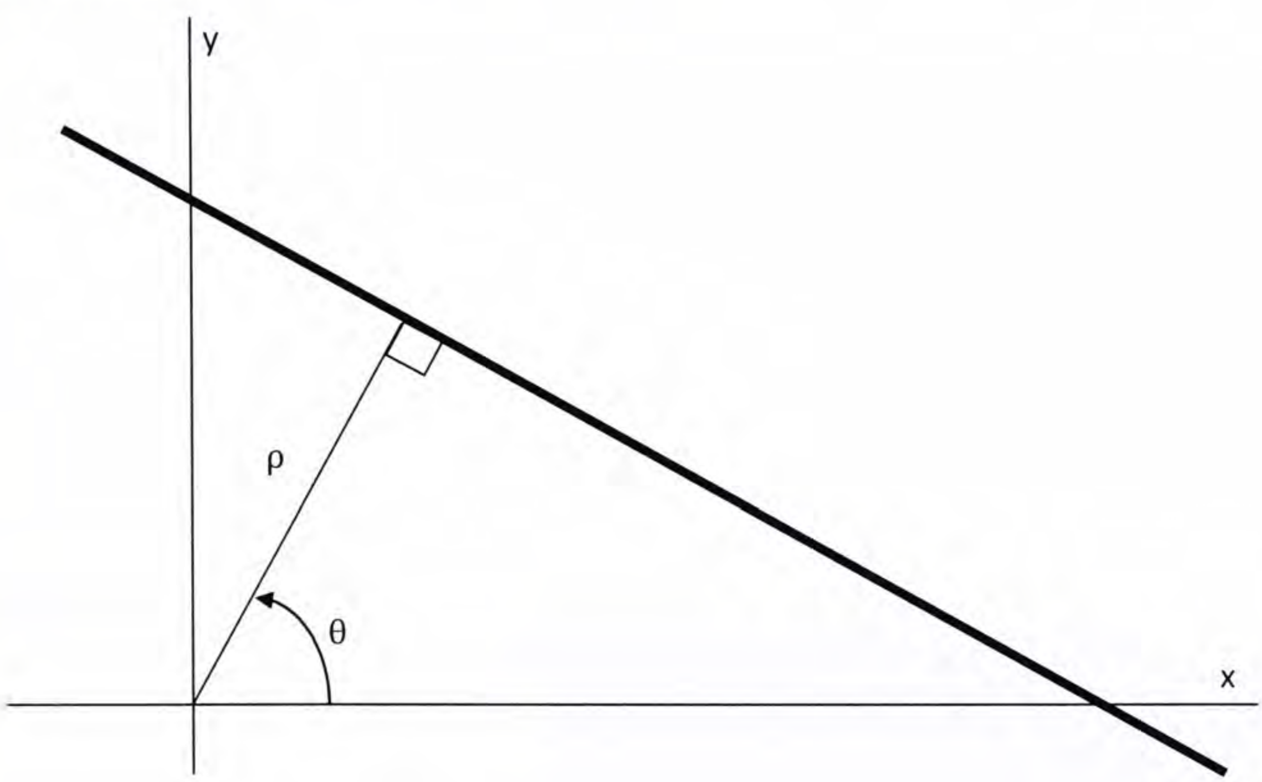

Figure 3.9. Parameters for a straight line

Generally, a straight line can be represented by y = mx + b which can be represented by (m,b) in parameter space. However, we can also converted a straight line into (ρ, θ) parameter space.

$$y = -\frac{\cos\theta}{\sin\theta}x + \frac{\rho}{\sin\theta}$$

where

ρ = shortest distance from the line to origin

θ = angle between x-axis and normal of the line

The above equation can be rearranged to

$$\rho = x\cos\theta + y\sin\theta$$

If θ is limited to $[0, \pi)$, any straight lines in x-y plane is unique in (ρ, θ) parameter space. While we have set of n points $\{(x_1,y_1), (x_2,y_2), \ldots, (x_i,y_i), \ldots, (x_n,y_n)\}$ in x-y space, we can transform all points to $\rho - \theta$ space such that

$$\rho = x_i\cos\theta + y_i\sin\theta$$

As all points on same straight line in x-y space will be transformed to same point on $\rho - \theta$ space, a straight line can be identified by finding an overlapped point in $\rho - \theta$ space (Figure 3.10)[15].

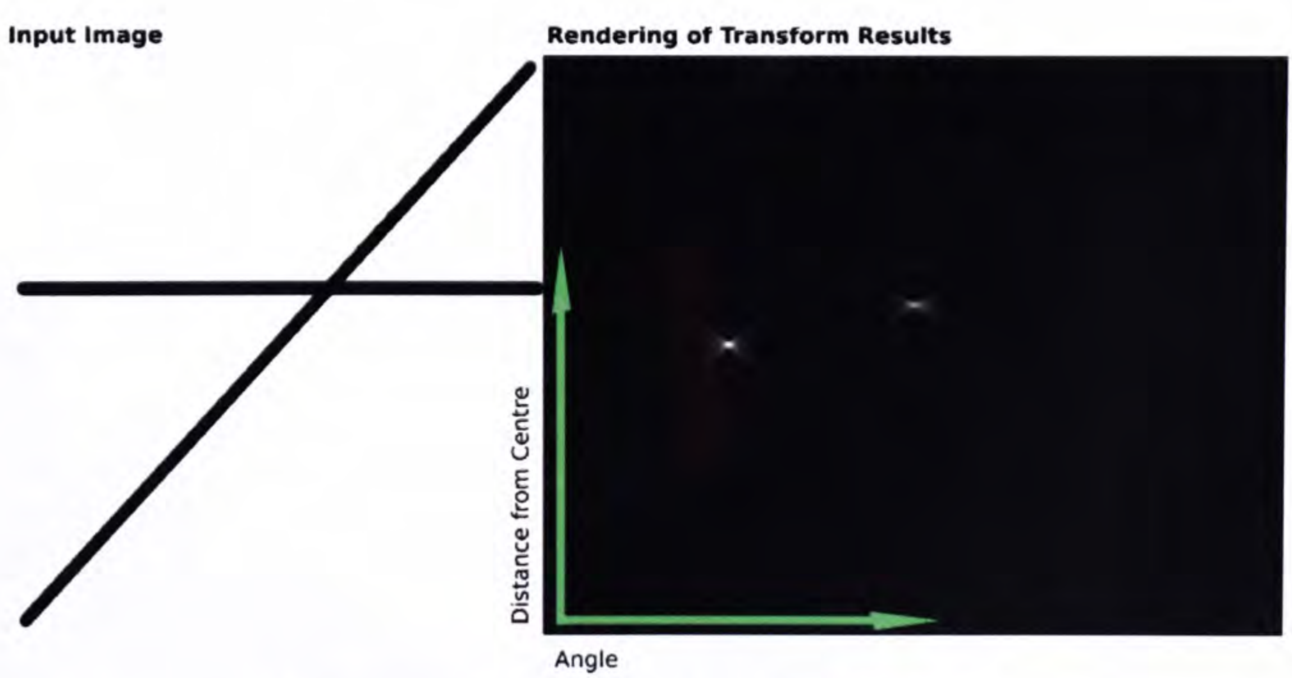

Figure 3.10. Example on Hough transform result

Hough transform is promising real-time image processing algorithm for detecting line information in an image [10][11]. Hough transform is used to recover white lines position in our project. As Hough transform algorithm takes into consideration of minimum line length and maximum gap between lines in the image, small area of noise or unwanted pattern can be filtered by setting relatively strict conditions to these parameters. Using this process, all point information generated by edge detector in the last step is transformed into useful line information regarding the end points of lines captured in the image.

3.3.5. Line analysis

With line information obtained by Hough transform, position of white lines on the image can be recovered. Because of the speed of the Segway Rider is limited, upper part of image is not needed for short term motion determination. Also, to avoid camera lens distortion affecting the determination of the program, ten pixels on each edge of the image are ignored.

To handle these two requirements, region of interest is set as the green area in Figure 3.11. This method has two advantages. First, with a smaller region of interest, processing power can be saved. Second, only two correct white lines can be seen in the image because angle of view of camera is known. Redundant lines found in the image can be ignored.

To recognize the existence of correct white lines in the image, minimum horizontal width in each row between correct lines (blue arrow in Figure 3.11) is pre-calculated. The width data can be found easily by simple trigonometry because the camera is fixed on the Segway Rider and all dimensions are measured.

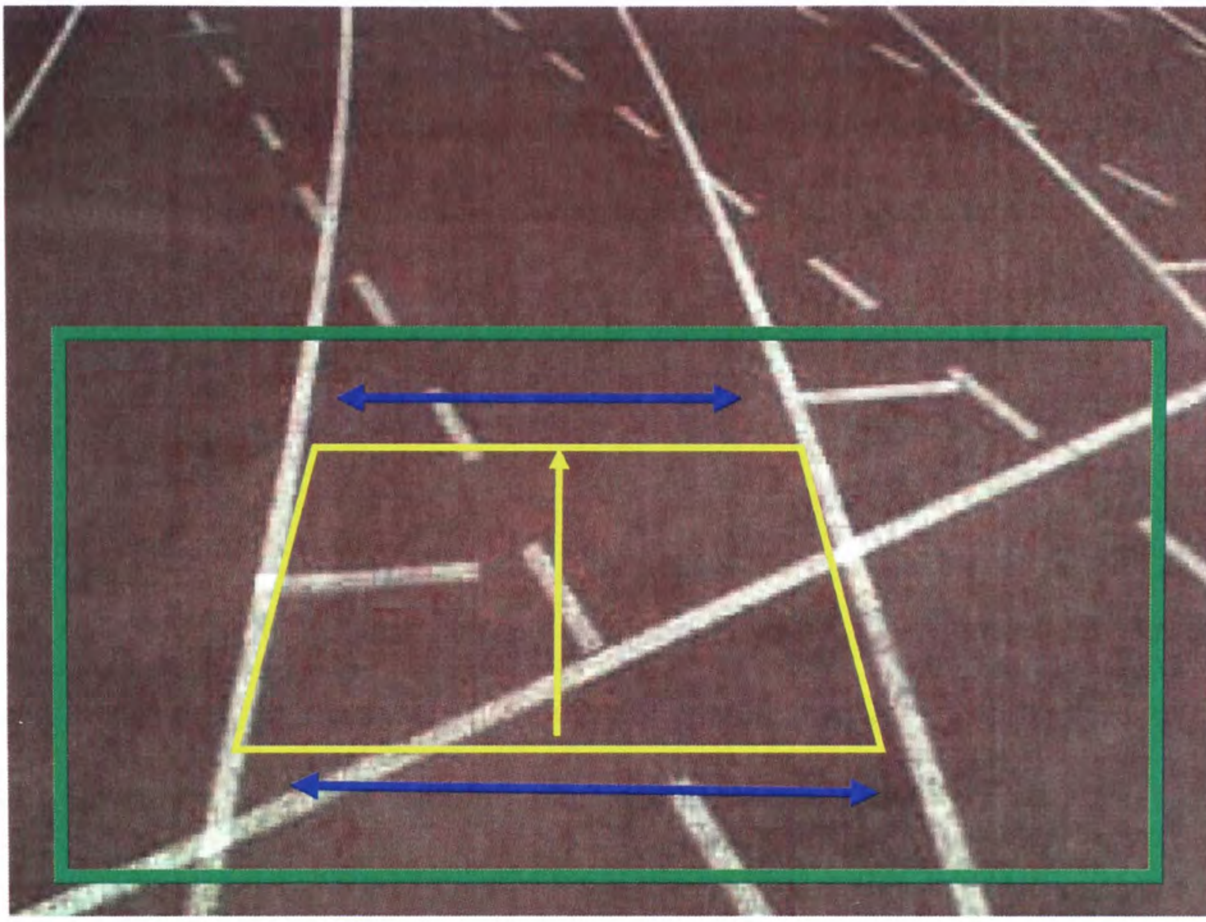

Figure 3.11. Region of interest

For each frame, the image is searched from the bottom of region of interest. Once a pair of correct line is found, the program tries to construct a trapezoid and its directional vector, yellow shown in Figure 3.11. This trapezoid is assumed to be the pointing direction of the Segway Rider. By calculating the angle between this direction vector and vertical axis of image, the angle shift of the Rider can be found. If the angle is larger than a threshold, the Rider should be commended to rotate towards a direction until the direction vector aligns the vertical axis.

3.4. Chapter summary

In this chapter, the grand challenge of this project is introduced. All processes for completing the objective are explained in detail. As a system uses computer vision only for auto-navigation, reasons and results of using these images processing algorithm are shown with real example captured from the real running.

The grand challenge was finished with a satisfying result. The Rider used 5 minutes and 25 seconds to finish the task (around 1.23 m/s), which was better than our own expectation. For safety reasons, we should not operate the Rider with higher speed in public environment without installing more safety equipment on the Rider.

Chapter 4

Stand and stay

4.1. Introduction

Standing or staying at a position is an important function for robotics mobility platform. For traditional mobility platform with three or more wheels equipped, it is easy to do so by just applying break on their wheels. However, for us, using Segway as our mobility base, it is not possible to add breaking mechanism on the machine itself. We need to use some special movement to achieve the function.

As there is no accelerometer or any velocity checking sensors installed on the Segway Rider, pure computer vision algorithm for this task is needed for the Rider. Our approach to solve the problem is observing the environment continuously. If there is large motion between frames, the Rider goes to opposite side in order to maintain the Rider on the same place. Also, the Rider should be able to stop in anywhere. However, there is no preset marker in the environment to signal it to stop. To solve this problem, we introduce a method to pick suitable marker in the environment randomly. All these will be described in following sections.

4.2. Box matching method

There are several methods to observe moving environment using computer vision, such as object matching and optical flow. However, considering the special moving behavior of the Segway Rider, with continuous "swing" motion, these methods have different problems to cope with. Due to the "swing" motion, if an object placed along the moving direction of the Rider is captured into the image, the object would move very fast across the image because of limited view angle of the camera. In order to solve this problem, only objects that are placed horizontally in the image should be taken for consideration. For this reason, typical optical flow or object matching cannot be directly applied to this project.

The box matching method, a tailor made method, is developed to fulfill the requirements. The method is extended based on basic object matching algorithm. Firstly, in order to cope with different environment, no previously specified markers should be used in our method. Markers are needed to pick automatically in the real time environment. Secondly, not all features found in the image can be used as marker. Horizontal features should be picked automatically. Features selecting algorithm is developed in the method. The logic flow of the box matching method is shown on Figure 4.1.

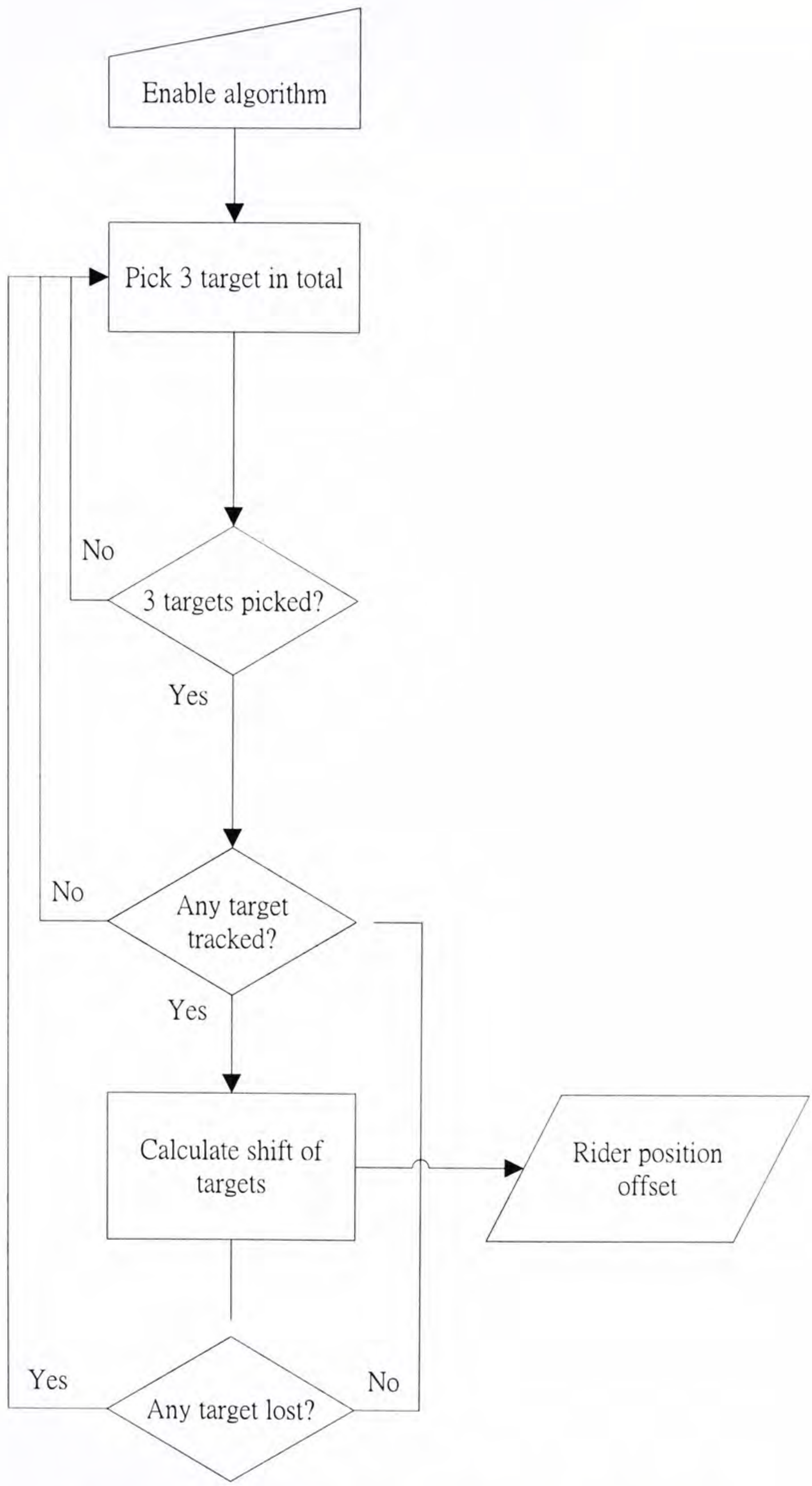

Figure 4.1. Flowchart of box matching method program flow

4.3. Image processing steps

There are totally five steps in the box matching method. The first three steps are used to find out suitable candidate features in the image, that is horizontal features. While marker is lost in the box matching step, candidate features are used to replace the lost one. The last two steps are used for matching and calculating the offset the Rider.

Step 1: Canny edge detection

Different from the steps we used in the grand challenge. Image in RGB space is converted into grayscale first. This is because the method is to be used in any environment, and not just color space as in grand challenge. Canny edge detector is then directly applied on the grayscale image.

After this step, point data of edges are gathered. In most cases, some noise data will be also contained in this set of points.

Step 2: Hough transform

Similar to the Hough transform we used in the grand challenge, this step tries to match all points obtained from edge detection and transforms these points into straight lines. Unlike the grand challenge, there will be many unknown objects found in this step.

After this step, large number of line segments can be found by Hough transform. Most small objects and noise can be filtered because shortest length of lines is considered in the process.

Step 3: Classify lines obtained

After applying Hough transformation, lots of lines is obtained. Due to the special moving behavior of the Segway Rider, which is keep on swinging forward and backward at low speed, objects located along the direction of movement oscillate in the view of camera. To avoid the random marker picked lose so frequent, only horizontal lines are taken into considerations and treated these lines as candidate.

After this step, only few horizontal line segments will be left. Refer to Figure 4.2, comparing with result image after edge detection, only obvious horizontal features will be taken into account.

Step 4: Pick marker randomly

Three random markers are taken for tracking in total. If marker is lost, the program needs to pick a line from the candidates randomly. Surrounded area of selected line is treated as new marker if no other marker exists nearby. If no suitable candidate exists, the program uses remaining marker to calculate result.

Step 5: Match box

This step is built based on basic object matching algorithm. For each frame, the program tries to match saved marker to the current frame. Planar offsets between markers original coordinate and current coordinate are used to take average for score calculation. The score is accumulated in certain ratio. The Rider uses the accumulated score to determine its next motion.

4.4. Experiment

Figure 4.2 shows screen capture for experiment of box matching method. The upper figure is initial frame and the lower figure is several frames afterward. The bold rectangles on top-left of figures are original coordinates of the marker. The thin rectangles with similar color are the matched marker coordinate in current frame. Inside the screen capture, image on top-left is real-time image, top-right is grayscale image, bottom-left is intermediate result after Canny edge detection, bottom-right is calculation result logging.

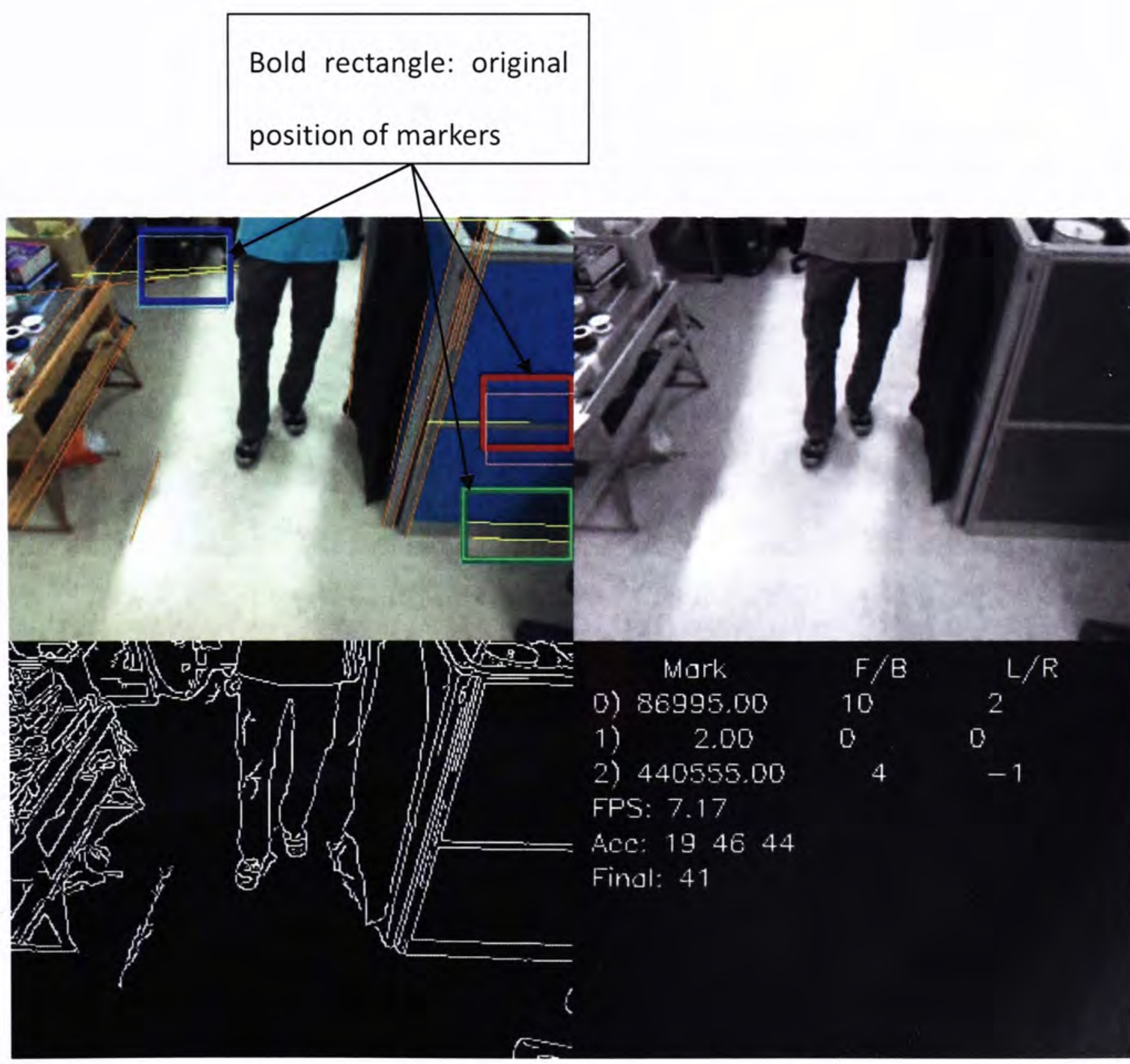

Figure 4.2.(a) Initial frame for box matching

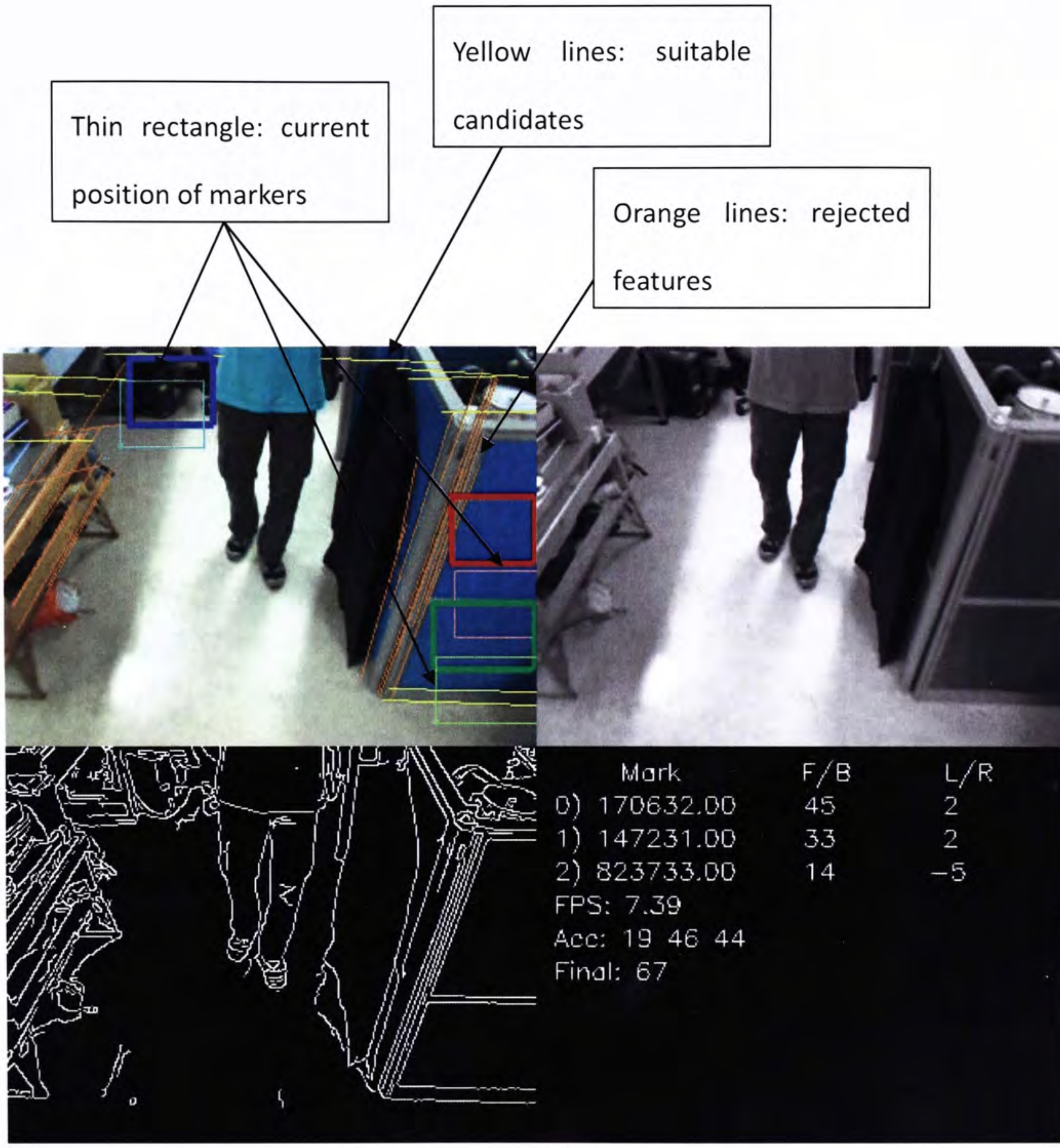

Figure 4.2.(b) several frames after Figure 4.2.(a)

4.5. Chapter summary

This chapter introduces the usage of box matching method. Image processing steps of the method is described in detail. A real experiment is demonstrated in this chapter.

It is difficult for the Segway to stay stably at a specific position. Currently, for the Segway Rider, it can keep itself within a range of 0.5 meter in several minutes. Since the errors in determining the Rider motion is accumulating, methods or techniques to reduce and reset the error accumulated are possible works.

Chapter 5

Conclusion and future works

5.1. Contributions

The thesis demonstrates the functions of the Segway Rider. Its functions are proven by the experiment, "the grand challenge". By using computer vision only, the Rider is successfully to complete the grand challenge of automatically running around the running track of the CUHK University Gymnasium.

In the development progress, different image processing methods are tested to choose suitable and reasonable process for the Rider.

On the other hand, since Segway PT does not have breaking function, it is necessary to implement stopping function on the Rider. A tailor made algorithm, box matching method, is developed to implement the standing function. Compute r vision process is described in detail.

The Segway Rider is an extensible robotics mobility platform. Space for adding new equipments is reserved in design. The Rider currently gives basic movement functions based on Segway PT. Well-tested vision system is installed on the platform. Radio frequency remote control is also available for the Rider.

5.2. Future works

Based on current setting of the Segway Rider, future researches can be developed in many ways. Firstly, robot control through internet becomes much popular in recent years because bandwidth of the internet is much larger and cheaper in cost. Real-time video feedback for remote access becomes possible. Secondly, indoor navigation using Segway Rider is suitable for narrow indoor environment in Hong Kong. There are prevailing interest I the construction of human scale robot for indoor environment. Segway's small footprint and zero turning radius features are perfect for such environment. Thirdly, 3D landscape scanning is also a possible research for the Rider. With its relatively slim dimensions, it can go everywhere conveniently which is not possible for traditional car based 3D scanning system.

Bibliography

[1] Kaustubh Pathak, JaumeFranch, and Sunil K. Agrawal, "Velocity and Position Control of a Wheeled Inverted Pendulum by Partial Feedback Linearization", IEEE Transactions on Robotics, Vol. 21, No. 3, June 2005.

[2] Segway official webpage. http://www.segway.com/.

[3] Hoa G. Nguyen; John Morrell; Katherine D. Mullens; Aaron B. Burmeister; Susan Miles; Nathan Farrington; Kari M. Thomas; Douglas W. Gage, "Segway Robotic Mobility Platform", SPIE Proc. 5609: Mobile Robots XVII.

[4] S. M. Goza, R. O. Ambrose, M. A. Diftler, and I. M. Spain, "Telepresence Control of the NASA/DARPA Robonaut on a Mobility Platform," *Conf. on Human Factors in Computing Systems (CHI2004),* Vienna, Austria, April 2004.

[5] R. O. Ambrose, R. T. Savely, S. M. Goza, P. Strawser, M. A. Diftler, I. Spain, and N. Radford, "Mobile Manipulation using NASA's Robonaut," *Proc. Int. Conf. on Robotics and Automation (ICRA'04),* New Orleans, LA, April-May 2004.

[6] R. Brooks, L. Aryananda, A. Edsinger, P. Fitzpatrick, C. C. Kemp, U. O'Reilley, E. Torres-Jara, P. Varshavskaya, and J. Weber, "Sensing and Manipulating Built-for-Human Environments," *Int. J. of Humanoid Robotics,* Vol. 1, No. 1 (2004).

[7] J. Searock, B. Browning, and M. Veloso, "Segway CM-RMP Robot Soccer Player," *Proc. RoboCup International Symposium,* Lisbon, Portugal, July 2004.

[8] Neurosciences Institute, *Brain-Based Soccer Playing Segway Platform,* URL: http://www.nsi.edu/nomad/segway/.

[9] M. Montemerlo and S. Thrun, "A Multi-Resolution Pyramid for Outdoor Robot Terrain Perception," *Proc. AAAI Nat. Conf. on Artificial Intelligence (AAAI'04),* San Jose, CA, July 2004.

[10] M. Nakanishi and **T.** Ogura, "A real-time CAM-based Hough transform algorithm and its performance evaluation," *ICPR'96,* vol. 11, pp. 516-521, Aug. 1996.

[11] Mamoru Nakanishi, Takeshi Ogura, "Real-time line extraction using a highly parallel Hough transform board,", *1997 International Conference on Image Processing (ICIP'97)* - Volume 2, 1997, pp582.

[12] Grasser, F., D'Arrigo, A., Colombi, S., and Rufer, A.C., 'JOE: a mobile, inverted pendulum', *IEEE Trans. Industrial Electronics*, 2002, 39, (1), pp. 107-114

[13] Brenna Argall, Brett Browning, Yang Gu, and Manuela Veloso. The First Segway Soccer Experience: Towards Peer-to-Peer Human-Robot Teams. In *Proceedings of the 2006 Conference on Human-Robot Interaction*, Salt Lake City, UT, March 2006

[14] Brenna Argall, Brett Browning, Yang Gu, and Manuela Veloso. The First Segway Soccer Experience: Towards Peer-to-Peer Human-Robot Teams. *Technical Report CMU-CS-05-161*, Computer Science Department, Carnegie Mellon University, 2005.

[15] http://en.wikipedia.org/wiki/Hough_transform

[16] Canny, J., A Computational Approach to Edge Detection, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8:679–714, 1986.

[17] Sobel, I., Feldman,G., "A 3x3 Isotropic Gradient Operator for Image Processing", *Stanford Artificial Project*, 1973.

[18] T. Lindeberg, "Edge detection and ridge detection with automatic scale selection", *International Journal of Computer Vision*, 30, 2, pages 117-154, 1998.

[19] Chih-Hui Chiu, Wen-Ru Tsai, Ming-Hung Chou, and Ya-Fu Peng, "Two-wheeled Robot Control Based on Self-tuning Output Recurrent CMAC", Proceedings of the International MultiConference of Engineers and Computer Scientists 2009, Vol I