On Instrumentation for Telecommands in Robotics

Mattias Bergström
On Instrumentation for Telecommands in Robotics

Mattias Bergström

June 2008
Abstract

This study is on instrumentation and testing of principles for controlling remote robots using telecommands. The three main topics studied are:

- Operator control/piloting of a mobile robot using the direction of a navigation camera. Thus the direction of the steering wheels are determined by the direction of the navigation camera.

- Control a mobile robot so that it navigates along a laser beam, a laser 'beam rider'. For detecting the laser beam a screen and a camera was used on board the robot.

- For the navigation camera to have a good view over the surrounding workspace it should be elevated in a high mast. This is essentially an inverted pendulum that was stabilizing using a propeller.

Keywords: Telecommands, navigation camera, laser beam rider, mast, video camera, time delay
Preface

This master's thesis is the final work in my Master of Science degree in robotics and mechatronics at Luleå University of Technology. The work has been performed at the Department of Computer Science and Electrical Engineering at Luleå University of Technology.

I would like to thank my instructor Åke Wernersson for guidance and the source of many ideas. I would also like to thank Sven Rönbäck and Håkan Fredriksson for help with instrumentation and ideas. Jerry Linblom was a great help for getting components ordered. To Rober Selber a great thanks for helping me getting started with \LaTeX{} which this report is written in. And last but not least my wife for putting up with many hours of work and incomprehensible rambling about robotics.
Contents

1 Introduction ................................................. 1
   1.1 Background ............................................. 1
   1.2 Content ................................................ 2

2 Visual feedback loops ........................................... 3
   2.1 Ugly .................................................... 3
   2.2 Ugly steering system .................................... 3
   2.3 Wiggly elevator ......................................... 4
   2.4 Wiggly docking ......................................... 4

3 Beam rider ...................................................... 9
   3.1 Introduction .............................................. 9
   3.2 Description of experiments 
     3.2.1 Detector ............................................. 9
     3.2.2 Picture processing .................................. 10
     3.2.3 Angle and bias detection ......................... 12
   3.3 Results and conclusion ................................ 12

4 Propeller stabilized mast ...................................... 15
   4.1 Hardware ............................................... 15
   4.2 Mechanics .............................................. 15
   4.3 Input module .......................................... 15
   4.4 Controller ............................................. 16
   4.5 Actuator module ...................................... 16
   4.6 Software .............................................. 17
   4.7 Input .................................................... 18
   4.8 Output .................................................. 18
   4.9 PID Module ............................................ 20
   4.10 Tuning the PID controller ............................. 20

5 Servo .......................................................... 23
   5.1 Servo standard ........................................ 23
     5.1.1 Control interface .................................. 23
     5.1.2 Feedback loop ..................................... 23
5.2 Testing
Chapter 1

Introduction

1.1 Background

The problem area behind this study is control of robots to operate, in workspaces at remote locations. The principal block diagram for control and sensing using telecommands is illustrated in Figure 1. The reader should observe the association box at the lower middle of the figure. In this box a decision is made, automatically or by the operator, on the identity of those objects in the scene that are relevant for controlling the robot. It is obvious from the block diagram that correct and reliable scene interpretation is essential for autonomous robots.

Figure 1.1: The basic block diagram of telecommands for control of robots and for sensing at remote locations. For different tasks, a mechanical system should be positioned relative to objects in the workspace. The internal loop gives the mechanics a smooth motion. The external geometrical loop is based on non-contact range measuring sensors. For a fully autonomous system the scene is also to be interpreted from the sensor signals i.e. each object is to be associated with the correct signal features. Essentially, using telecommands this loop is closed when the operator selects the association switch.
1.2 Content

The three topics in this study are as follows:

In chapter 2, we study how a turn able navigation camera can improve the operation of a remotely piloted robot vehicle. Compared with a fixed camera, the reason for the improvement is that we get one layer deeper in the control scheme.

In chapter 3, we study how a laser beam can be used for control of a robot. Essentially, the robot can navigate along the laser beam. We have a beam rider.

In dense cluttered work spaces the navigation camera should be lifted up to get a better view. The topic in chapter 4 is using a propeller for stabilizing a high mast.
Chapter 2

Visual feedback loops

Detta kapitel kommer att ge en inblick i det arbete som har utför kring två fjärrstyrd enheter Ugly och Wiggly. [1]

2.1 Ugly

The Ugly robot is essentially, a remotely piloted RC-car but with a special steering. The directions of the navigation camera and the steering wheels are the same, apart from a gain factor. This is instrumented by having the two servo motors driven from the same output of the RC receiver. The block diagram and camera setup is given in figure 2.1 and 2.2.

2.2 Ugly steering system

Ugly setup is of an ordinary RC-car, but has also a video transmitter that sends the video to the operator. [3]

![Diagram of Ugly robot with navigation camera at angle $\beta$ and steering wheel angle $\alpha$.](image)

*Figure 2.1: Principle drawing of the robot with navigation camera at angle $\beta$ and steering wheel angle $\alpha$.***
2.3 Wiggly elevator

The experiments conducted here where to test the possibility for making a task of while using results from the Ugly in chapter 2.7. The task that Wiggly where used for during the elevator test was to hit the elevator door button, the door opens automatically, and then maneuver Wiggly inside the elevator cart. The setup can be seen in figure 2.3.

The testing of laser enhancement in the picture is done by lasers!

2.4 Wiggly docking

To further test the precision of steering via telecommand and the a docking test were performed a the pushing rod of Wiggly where replaced with a forklift construction see figure 2.7. The forklift where able to lift a small block and and needs mush higher accuracy than the test with the push button from section 2.3.
Figure 2.3: Setup of the elevator push button experiment.

Figure 2.4: Laser reference lines.
Figure 2.5: Operator view.

Figure 2.6: Wiggly entering the elevator.
Figure 2.7: Wiggly with the forklift for docking operations.
Chapter 3

Beam rider

In this chapter we describe instrumentation for detecting and finding the location of a red laser beam. It is also tested how the robot can navigate along the laser beam. The detector is built up using consumer electronics and all data computation is made in MATLAB[1].

3.1 Introduction

The goal is to detect a laser beam with a webcam so that a robot can navigate along the beam. It should be possible to get a fast moving robot with relative low errors in position. It was also a test of what can be done with cheap consumer product such as a webcam.

3.2 Description of experiments

3.2.1 Detector

To detect the laser beam two transparent plastic films where placed parallel to each other. Then a webcam (Philips Toucam pro II) were used to take a picture of the films from below. The picture shows light that is emitted from the plastic when the laser beam hits the plastic films. The setup can be seen in figure 3.1

To get a better contrast of the red laser beam a box was made covered in blue plastic that filtered out the red light from other disturbing sources. The strong intensity of the laser beam is so intense that it will penetrated the plastic films leaving a red dot in a blue environment figure 3.3 shows the detector whit a laser beam going through it. The film that was used is from rosco and the superfel series. Figure 3.2

There is also some ordinary white paper on top of the detector. This is to shield the camera from locking directly into lamps in the sealing and go into blooming. The camera is put in to automatic mode for the light intensity
modulation. This is so the sensitivity of the camera is at its maximum independent of the light in the surrounding environment. There is also blue film under the white paper. All the surfaces that the camera is seeing is covered whit blue film.

3.2.2 Picture processing

For processing the pictures taken by the camera MATLAB was used with the Image Computing Toolbox and Image Acquisition Toolbox. When getting a picture from the camera the red dot is already clearly visible due to the blue film. The whole idea with the film is to make a so good picture from the beginning so no image processing is needed to get the laser dots to be seen clearly. The only thing that is done to the image is to threshold it to find the position of the laser dots. The main reason for this passive filtering is to avoid lots of computation because MATLAB has very poor real time ability. The picture from the camera comes to MATLAB in form of YUV coding. This is three matrixes that are of the same size as the picture there

Figure 3.1: To get a measurement of the sensor position towards the laser beam a web-cam locks at the red dots on the plastic film

Figure 3.2: Characteristics of the blue film used for the detector.
3.2 Description of experiments

Figure 3.3: Sensor setup with a laser beam going through the detector.

...the first is the intensity of the picture and the other two is the blue and red chrominance.

Figure 3.4: From the webcam picture the red chrominance buffer is extracted (middle picture). After thresholding a binary picture is obtained from which the position of the dots are calculated.

...in figure 3.4 is the first image what the camera sees. To filter out the red dots the intensity of the red chrominance buffer is look at. In the second image the peaks are quite distinguished from the background noise, so the threshold is just set a bit lower then the maximum value of the peaks.
3.2.3 Angle and bias detection

To get the positions of the dots the mean value was taken for the x-axis position over all the pixels for each of the dots. When the position is computed based on the pixel values they have to be scaled into meters in this case. To calculate the resolution \( R \), where \( W \) is the width of the sensor and \( P \) is the number of pixels on that length equation (3.1) was used.

\[
R = \frac{W}{P}
\]  

(3.1)

If the values are just scaled the zero position will be in one of the corners so a bias is added to get the zero in the middle of the detector. When all this is computed the output from the program is one position for each side of the detector, so the bias is just the calculated position on ether on of the sensor sides. To calculate the angle between the sensor and the laser beam the difference between the two dots position is used as in equation (3.2), where \( L \) is the length between the plastic films in meters, and is the position of the dots in meters and is then the angle between the sensor and the beam.

\[
\alpha = \tan \frac{x_1 - x_2}{L}
\]

(3.2)

With equation (3.2) the angle difference will be in the sensors coordinate system.

3.3 Results and conclusion

The program was able to work in 10 fps although sometime the calculations tuck longer time. The resolution in the sensor was 0.39mm per pixel, with a detector width of 78mm This gives an angle resolution of approximately 5mrad. These measurements are true in theory although there is some noise in the picture which will make the detected dot to change form randomly with approximately one pixel in the edges under normal light. If the light is poor the noise levels will rise and result in a dot which changes form more than in good light. There is also the fact that the camera works in half the resolution in the chrominance buffers but due to the mean value calculation it will give a proximal resolution that is the same as the intensity. A big problem that occur trying to detect the laser beam is the shifting light while traveling through corridors. The best way to deal whit this was through the blue film, trying to detect the laser directly on a whit transparent plastic film gave to little light from the laser and weary match interference from red objects in the room. Increasing the laser intensity gave little result and the dot become scattered and big as a picture. For the blue box to work properly all surfaces inside the box needed to be blue. If any surface had a other color
than blue, the camera locked the red chrominance on that surface and the dot no longer become the most intense region of the picture.
Chapter 4

Propeller stabilized mast

In dense cluttered work spaces the navigation camera should be lifted up to get a better view. The topic in this chapter is using a propeller for stabilizing a high mast.

4.1 Hardware

The hardware consist of three modules, input module, controlling module, actuators module and mechanics.

4.2 Mechanics

The mechanics consist of ground platform (with battery, USB-RS232 module, charger, angle sensor and joint) and the mast with the mounted motor and propeller on the top. The length of the mast can vary from one meter up to few meters. During the controller developing the mast was one and a half meter long.

4.3 Input module

I am using the Gill’s Blade 360 degree rotary position sensor which use inductive method for measuring angle. I had some problems with it because the power supply wire for the motor was attached too close. The switching motor controller produces high current peaks which affected the precision of angle measurement. I solved the problem by simply removing the power supply wires far away from the joint where the angle sensor is mounted. Using the sensor’s RS232 interface made a simple and reliable reading of sensors value. The USB to serial cable is used to connect it to a computer. Ground part of the controller
4.4 Controller

The normal laptop PC running Windows XP with the USB-to-RS232 cable was used as a controller. The controlling software is written in C++ with the QT framework, so that the running platform is in dependable (it can be run also on embedded system running Linux).

4.5 Actuator module

Actuator module consist of a simple RS232 driven servo controller (which can handle up to eight servos), brush less engine controller (which can be controlled using servo controller), one mini servo motor (for the rotor blade
Figure 4.2: Block diagram of used modules.

Figure 4.3: Block diagram of used modules.

pitch control) and brush less motor with hollow shaft and mounted variable pitch propeller. Module mounted on the top of the mast

4.6 Software

Similar like the hardware the software consist of three modules input, PID controlling algorithm and output. Two generic classes are used to provide platform undependable serial port interface called QextSerialPort and configuration file parser class called ReadFile. The user graphical interface is included in class GUIWidget. Once the controller is tuned, the PID pa-
rameters can be hard coded and the interface can be removed without any necessary changes to other parts of the code. User Graphical Interface

4.7 Input

The class that takes care of the input is in the Input.h class. Currently the only input parameter that I can use is angle of the mast regarding to the ground. Because used angle sensor is using asynchronous communication I use a timer driven method to send a periodic request to the angle sensor serial interface. The method is called sendRequest(). The second method in this class is called readValue() and is also periodically called by the timer. This reads the output string from the angle sensor and tries to parse the output string which usually looks like this: "+0342.0122
" If the string is successfully parsed the method emits the angleSignal(float) signal which is directly connected to a PID class. In order to decrease the unwanted delay I tuned the timers to get as much samples as I could. The maximum sample rate that I have got is 500 samples per second. To eliminate the noise I am using sliding window average filter. This way I have got 100 stable angle reading.

4.8 Output

The output class must handle two parameters: rotor blade pitch and motor power. Those parameters are passed by two main methods. The first one
is called setMotorPower(float). This method gets the motor power as float argument expressed in percentage from 0 recalculated using the minimum and maximum servo controller offsets and sent as a four bytes string to the serial port driven servo controller. The first byte has a 0xff value, the second byte of the string is the servo number, the third one is the value/position of the servo and the last one is carriage return character. The second method is called setPitch(float). This method gets the requested pitch as float argument expressed in percentage from -100 value is then recalculated using the minimum and maximum servo controller offsets and sent as a four bytes string.
string to the serial port driven servo controller. The first byte has a 0xff value, the second byte of the string is the servo number, the third one is the value/position of the servo and the last one is carriage return character. The third method setOutputPower(float power) that was first planned to use as an output which would automatically compensate the propeller pitch and the motor in order to get a constant speed of the rotor blade. When I found out that the rotations speed doesn’t vary too much I decided not to use this method. To get a total linear response I would need the blade rotation speed as well. Other methods in this class are used only to provide information to a GUI class.

4.9 PID Module

The module uses a basic PID controller algorithm. The input feedback value is received by the getFeedBack(float) method. This method just stores the value into a PID’s private feed back variable. The timer calls the calculate() method periodically 100 times per second. Each time the method calculate the the error value between the feedback input variable and reference variable (which can be also set using the GUI input reference slide bar). Beside the error, the speed of the mast calculated using the previous and current mast positions. User can tune the controller with changing three main controller parameters P, I and D using the GUI. The final output is then calculated using the error value, speed of the mast and P, I and D parameters using the following equation:

\[ \text{Output value} = P \times \text{error} + I \times (\text{summed previous errors}) + D \times \text{mast speed} \]

The calculated output value is then emitted as a outputSignal(float) signal to an output class. To see how the controller was tuned see "Tuning PID Controller" paragraph. Other methods are used only to interfere to a GUI module.

4.10 Tuning the PID controller

The tuning starts with zero pitch setting. We start the controller program and turn on the motor by sliding the motor output slider to 80% change the offset pitch offset in the code itself in order to achieve zero force on the mast when the motor is running. We have to do this procedure only when we assemble the mechanic. Later we leave this constant alone. After we start to tune the PID controller itself. First we set all P, I and D parameters to zero. Then we turn on the motor (usually I set it to 80% point we shouldn’t feel any force form the propeller if we move the mast from the floor to the standing (or zero degree) position. Then we start to tune the P parameter. We hold the mast in +/- 20 degree position and then we start to increase the P value. If everything works properly the mast should be able to get in
Figure 4.6: Block diagram of software modules.

a zero position itself. If the P value is too big we can see this as an overshoot. If the P value is too small the mast can not reach the zero position. Once you achieved that the mast return itself to zero position (at this point it is not stable and still oscillates around zero) you can start tuning the D component. This component is continuously trying to set the mast speed to zero. Increase the value until you the mast stops to oscillate. Now the mast should be stable but probably slightly off the zero value. This can be fixed with carefully increasing the I parameter. After this procedure you should have stabilized mast. During this procedure you must some how protect the mast from hitting the floor or any other object, because it can really easily became very unstab
Chapter 5

Servo

The development of radio controlled models servos\[2\] have the ... RC servo is used in many applications for its simple interface and can be purchase in a wide range of ... from low price to high end servos.

5.1 Servo standard

5.1.1 Control interface

The RC servo has a simple on wire PWM interface for controlling the device ... the length of an positiv pulse sets the servo position. To get the servo in the middle position a 1.5 ms is required and for the end positions the signal has a span from 1.0 - 2.0 ms, in resent years the span has increased in some servos to 0.9 - 2.1 ms for better resolution. skriv omj uppdateringshastighet

5.1.2 Feedback loop

In early RC servos the feedback loop was analog and dun by converting the pulse signal to a voltage and compared with the voltage from a potentiometer that is attached to the outgoing shaft.

![Block diagram for a analog RC servo.](image)

**Figure 5.1:** Block diagram for a analog RC servo.
## 5.2 Testing

<table>
<thead>
<tr>
<th>Specification</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTROL SYSTEM</td>
<td>+PULSE WIDTH CONTROL 1500\mu ssec NEUTRAL</td>
</tr>
<tr>
<td>VOLTAGE RANGE</td>
<td>4.8V TO 6.0V</td>
</tr>
<tr>
<td>TEST VOLTAGE</td>
<td>AT 4.8V AT 6.0V</td>
</tr>
<tr>
<td>OPERATING SPEED</td>
<td>0.23sec/60° NO LOAD 0.18sec/60° NO LOAD</td>
</tr>
<tr>
<td>STALL TORQUE</td>
<td>4.4kg.cm(61.10oz.in) 5.5kg.cm(76.37oz.in)</td>
</tr>
<tr>
<td>IDLE CURRENT</td>
<td>3mA AT STOPPED 3mA AT STOPPED</td>
</tr>
<tr>
<td>RUNNING CURRENT</td>
<td>160mA/60° NO LOAD 180mA/60° NO LOAD</td>
</tr>
<tr>
<td>STALL CURRENT</td>
<td>900mA 1100mA</td>
</tr>
<tr>
<td>DEAD BAND WIDTH</td>
<td>1\mu ssec 1\mu ssec</td>
</tr>
<tr>
<td>OPERATING TRAVEL</td>
<td>40° /ONE SIDE PULSE TRAVELING 400\mu ssec</td>
</tr>
<tr>
<td>DIRECTION</td>
<td>CLOCK WISE/PULSE TRAVELING 1500 TO 1900\mu ssec</td>
</tr>
<tr>
<td>MOTOR TYPE</td>
<td>CORED METAL BRUSH</td>
</tr>
<tr>
<td>POTENTIOMETER TYPE</td>
<td>6 SLIDER/INDIRECT DRIVE</td>
</tr>
<tr>
<td>AMPLIFIER TYPE</td>
<td>DIGITAL CONTROLLER &amp; MOSFET DRIVER</td>
</tr>
<tr>
<td>DIMENSIONS</td>
<td>38.8x19.8x36mm(1.52x0.77x1.41in)</td>
</tr>
<tr>
<td>WEIGHT</td>
<td>40g(1.41oz)</td>
</tr>
<tr>
<td>BALL BEARING</td>
<td>DUAL/MR106</td>
</tr>
<tr>
<td>GEAR MATERIAL</td>
<td>HEAVY DUTY RESIN</td>
</tr>
</tbody>
</table>

*Table 5.1: Specifications for Hitec HS-5475HB Standard digital servo*
References


