OBJECT TRACKING AND SERVO CONTROL OF AUTONOMOUS MOBILE ROBOT

Sathya Govindarajan
Senior Lead
Automatic Data Processing, New Jersey, USA
E-Mail: sathyag413@gmail.com

A Double Five Star Paper

Abstract

The work presented in this paper is concerned with the problem of finding a feasible method for real-time tracking of a target (object) by an Autonomous Mobile Robot (AMR) using Machine Vision, Image Processing and Pattern Recognition and Tracking techniques. The major objective of the work reported in this paper is to develop a system for 'tracking of a specific object based on colour information' and it is achieved using a Real-Time Machine Vision Based Tracking Robot system consisting of a Parallax Boe-Bot Robot Kit, Basic Stamp 2 microcontroller, Board of Education, Robot's aluminium body, two Parallax servo motors, four 1.5V AA sized batteries, a RS232 serial cable, and a wired USB web camera. The software tools that are used in developing this system are Visual Studio .NET 2008 with MSDN, DirectShow.NET API with AForge.NET wrapper, Webcam device driver, and Driver for USB to RS-232 converter. This paper provides a comprehensive composition of the results achieved due to a rigorous research work carried out by the author.

Keywords: AMR, Machine Vision, Target Tracking

1. Introduction

This paper deals with a novel object tracking methodology using color information and control strategy of servo motors using Basic Stamp 2 microcontroller. Object tracking is done on the live video that shoots a real scene. Object tracking stage comes after the image processing and pattern recognition stage where the video frames are grabbed and processed. The object tracking module tracks the target in a two dimensional co-ordinate system and returns (updates) the spatial co-ordinates of the object to be tracked, which specifies its current position of the object in a frame.

1.1 Object tracking types

Different types of object tracking techniques as found in standard literature are listed below.

- Mechanical trackers that are based on a kinematic structure (either serial or parallel), which consists of links interconnected with sensorized joints.
- Inertia trackers, which are devices that consist of gryroscopes and accelerators and measure the rate of change of the translation velocity and the angular velocity and the angular velocity of an object.
- Ultrasonic trackers, where transmission and sensing of ultrasonic waves is used. The time taken for a brief ultrasonic pulse to travel from a stationary receiver placed in the environment to a receiver attached to the moving object is measured and used to identify the object's position.
- Magnetic trackers, which are noncontact devices that use the magnetic field produced by a stationary transmitter to measure the real-time position of a receiver placed on the moving object.
- Radio and microwave trackers, where the time of flight of the corresponding type of waves from a stationary transmitter to a moving receiver on the object of interest is measured to determine the range of an object.
- Thermal trackers, based on the heat signature of the target object, track its position. This can be extended to thermal imaging which can be used in the case of machine vision tracking.
- Machine vision tracking uses video as input, processes different video frames by using various image processing techniques, extracts the features, segments the object from the background and calculates the position of the object in the video frame.
- Hybrid trackers, which use more than one of the above position management technologies to track objects more accurately than a single technology would allow.

This paper mainly discusses machine vision-based techniques where tracking is carried out using video frame inputs and extracting the centroid positions in every frame.

1.2 Major problems associated with tracking

The first step for moving object detection and tracking is concerned with the problem of separating foreground from background in an image. Essentially, the background has to be removed from each frame of a video sequence in order to fix up the tracking point in an object in the foreground. For example, separating a moving man or a moving car in a given scene is known as foreground detection. The problem complexity in extracting the moving objects in a dynamic scene, however, increases with the increasing presence of motion due to various other phenomena. For example, factors such as the change in illumination, variation of sunlight as the day progresses, change of appearance of colour due to white balancing (and/or colour correction) in the colour digital camera itself, change of pixel values due to flickering and other phenomena, create false motions in the dynamic scene. Also motions due to shadow movement, tree movement, and many such other events in the scene need to be considered and detected as false motion. In view of the above, we need to model the background continuously so that only moving objects of interest may be detected and tracked accurately.

In this paper, separation of foreground from background is done at the image processing level itself. That is, the object in a frame is segmented in the form of a binary image from the background with the help of colour filtering. The object is represented with white pixels and background is represented as black pixels. Thus, it becomes easy for the AMR to track the object of interest neglecting the background.

1.3 Model of a tracking system

A tracking system is usually modelled as a three-state sequential machine. The sequential machine has three states, i.e., *locking*, *tracking* and *recovery*. Functions of each state are explained below:

- ❖*Locking state*: Initially the system is in locking state that is, when the camera is in search mode, searching for targets. During this state the processing is carried out on the whole image frame. The system will partition the image frame captured by camera into a number of moving objects. The history of these objects is extracted by checking the trajectory followed by the objects, and confirmation of the moving object is carried out in automatic mode. Once the target is confirmed the control of the system is transferred to tracking state.

- ❖*Tracking state*: This stage should use computationally inexpensive techniques. Current location extracted by locking state is used for processing. Next position of the target is identified, and that positional information is stored in history database. If the target does not exist in the predicted window area, then the system control is transferred to recovery state.

- ❖*Recovery state*: Quite often the moving object of interest may be lost temporarily or permanently. In this state if the target is lost, the system will try to recover the target from low-resolution image. If the target is recovered in a few frames, then the system will transfer control to tracking state; otherwise, it remains in recovery state till its predefined time expires. After the time is elapsed, control transfers to locking state.

In this paper, we elaborate the techniques of tracking objects in a two-dimensional plane.

1.4 Two-dimensional object tracking

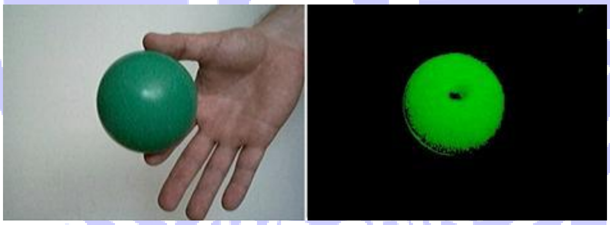
Two-dimensional rigid object tracking tries to determine the motion of one or more rigid objects in the image plane (i.e. its position over time instants). The image plane is induced by the relative motion between the viewing camera and the observed scene. *A basic assumption behind 2-D rigid object tracking is that there is only one, rigid, relative motion between the camera and the observed scene. This is the case which is realized in this work*. This assumption rules out flexible objects (that is articulated objects) like a moving human body, or deformable objects like a piece of cloth. Methods for 2-D rigid object tracking can be classified in different categories according to the tools that are used in the tracking: (i) Region-based methods, (ii) Contour-based methods, (iii) Feature-based methods and (iv) Template-based methods. Two-dimensional rigid object tracking methods constitute the basic building blocks for other categories of tracking algorithms. For example, an articulated object tracking algorithm may include a rigid object tracking module in order to track the rigid parts that make up the articulated structure.

1.5 Region based object tracking

It is usually an efficient way to interpret and analyze motion observed in a video sequence. An image "region" can be defined as a set of pixels having homogenous (similar) characteristics. It can be derived by image segmentation, which can be based on distinctive object features like colour, edge, etc, and on the motion observed in the frames of a video sequence. Essentially, a "region" would be the image area covered by the projection of the object of interest onto the image plane. Alternatively, a region can be the bounding box or the convex hull of the projected object under examination.

Colour information proved to be very effective in region based object tracking because it enables fast processing while providing results robust enough to perform real time tracking. Colour segmentation is the core of colour based object tracking algorithms. If the colour of the object to be tracked can be modelled efficiently and distinguished from the colour of other objects in the scene and the colour of the background, it can be very useful tracking cue, combined of course with appropriate colour similarity metric. The major problem in colour segmentation and tracking is to provide for robustness again illumination changes. This can be achieved for example, by controlling the illumination conditions, which is of course, impossible in the real world environments especially outdoor scenes. Alternatively, illumination-invariance or colour correction can be used. The former aims at representing colour information in a way that is invariant to illumination changes, whereas the latter attempts to map the colour responses of a camera obtained under unknown illumination conditions to illumination independent descriptors. For example, one common way to obtain illumination invariance is by using only the chromaticity values in a suitable colour space (example: the chromaticity components H, S in the Hue-Saturation-Value space). Alternatively, one can normalize the colour space. For example, in the Red, Green and Blue (RGB) space, this would mean that instead of using the RG and B component values, one can use value R/(R+G+B), G/(R+G+B), and B/(R+G+B) respectively or the value R/max(G,B), G/max(R,B), R/max(R,G). For colour constancy, a number of well-known algorithms such as variants of Grey World Algorithm, colour by correlation and so forth can be applied.

Many tracking algorithms focus on tracking humans (the whole body or body parts such as, hands, face, etc.). It becomes obvious that, in this case, the distinctive colour of the human skin can serve as an appropriate means of locating and tracking people in the video. A colour segmentation algorithm can be devised in three steps, namely the choice of a suitable colour space, the modelling of the skin, or colour distribution over selected colour space and the method used to classify the individual pixels to object (skin) and non-object (non skin) pixels. The most important part now is the segmentation. Once having found the colour of an object, it has to be segmented. Segmentation means that we are going to clearly distinguish the object from the background. Thresholding is one of the most primary segmentation techniques. The output of thresholding would be a binary image where the object is represented in white pixels and the rest in black pixels. Now it is easy to identify the object's position by calculating the centroid of all the white pixel value. Essentially, I made use of Rajan Transform based iterated thinning algorithm to find out the centroid of the object. This step is repeated for each and every frame and a "track" of the object is formed based on the locus of all the centroids formed from each frame. Figure 1.5.1(a) shows a green ball (I am holding it) and (b) shows its separation from the background.



(a) (b)

Figure 1.5.1(a): Green ball from a still frame of a video and
(b): Colour-filtered and thresholded frame highlighting the ball, the object of interest

2. Servo Controlling

Every robot has got to have some actuator or the other, so that it can manoeuvre itself. An actuator is a mechanical transducer for moving or controlling a mechanism or system. There are many types of actuators. Some of them are given below: (i) Electrical motors (stepper and servo motors), (ii) Pneumatic actuators, (iii) Hydraulic pistons, (iv) Relays, comb drive, (v) Piezoelectric actuators, (vi) Thermal bimorphs, (vii) Digital micro mirror devices and (viii) Electro active polymers.

2.1 Servo motor

Like DC motors, servo motors have been around a long time and are used in a wide variety of applications. The basic design of the servo motor is the same no matter the size or manufacturer. Figure 2.1.1 illustrates a cut-away of a servo and shows us what it looks like on the inside. The bottom has the servo controller board which interprets the signal input and turns on the motor inside which is geared up. We cannot see it clearly but there is also a potentiometer (also known as a variable resistor) inside the servo as well. The servo uses the potentiometer for finding the different angles that it will rotate to. The servomotors that I have used in this work are Parallax servo motors (Refer to figure 2.1.1(b))

(a) (b) (c)

Figure 2.1.1(a): Servo motor; (b): Parallax servo motor; (c): Servomotor connection

Parallax motors have three wires coming out of them. The power and ground wires are hooked directly up to whatever battery or power supply is used to power the servos. The Signal wire (white) will be hooked up to the microcontroller used to control the servo, in our case the Basic Stamp 2. A noticeable first impression is that the servo requires only one pin from the microcontroller. The signal that we need to create in order to control the servos is called a *Pulse Width Modulation* signal or PWM for short. The general requirements are: (i) Frequency: 50 Hz, (ii) Up-time: 0.9 mS - 2.1 mS and (iii) Down-time: 19.1 mS - 17.9 mS. At first glance these definitions and numbers might make little or no sense. So let us look at a simple PWM wave at 50Hz.

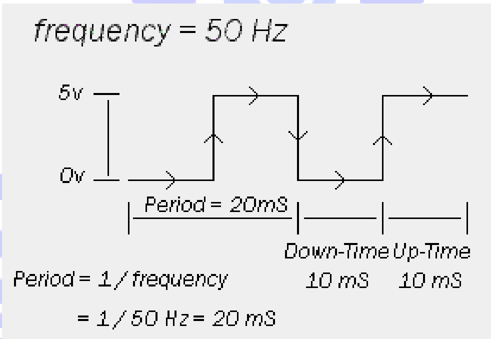


Figure 2.1.2: PWM wave form

PWM wave is just a signal that changes between 0 volts & 5 volts (digital logic 0 and 1). We see that the wave is symmetrical; uptime is 10mS & downtime is 10mS which when added together give us the period (10mS + 10mS = 20mS). Now that we have a feeling for what a PWM signal looks like, let us change the wave to look like a normal servo signal input. Frequency: 50 Hz, Up-time: 1.5 mS and Down-time: 18.5 mS

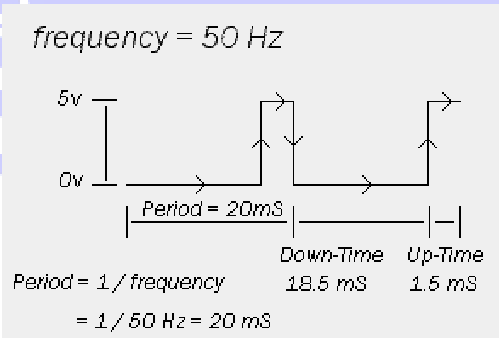


Figure 2.1.3: PWM signal for servo

The Up-time is what determines the angle the servo motor moves to. Since standard servos can rotate up to 90°, the different range of possible up-time values will equal a certain angle. Here are some examples:

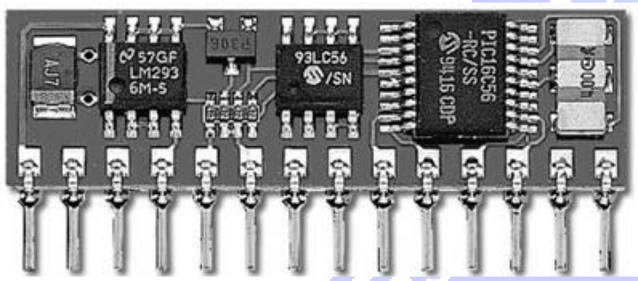
Up-Time = 1.0mS Period = 20mS Angle = 0°	Up-Time = 1.5mS Period = 20mS Angle = 45°	Up-Time = 2.0mS Period = 20mS Angle = 90°

2.3 Introduction to Basic Stamp 2 microcontroller

Generally, a microcontroller is a mini-computer used to control electronic devices. Microcontrollers today are used in countless powerful applications. Among the typical applications of a microcontroller to-date include washing machines, cars, mobile phones and robots. Originally, microcontrollers were only programmed in assembly language, or later in C codes. These programming languages however, require technical understanding of the entire system and its programming. Recently, some microcontrollers have begun to include a built-in high-level programming language interpreter for greater ease of use. BASIC is a common choice, and is used for programming the popular BASIC Stamp Microcontrollers. Today, robotic developers prefer the Basic Stamp range over other common microcontrollers due to its operation simplicity. Basic Stamps are also used in many universities and colleges as a teaching tool to equip students with the knowledge of microcontrollers and its applications, typically robotics.

Basic Stamp 1 (BS1)

The BASIC STAMP 1 (BS1) was the first of the STAMP series, using a mini version of BASIC or BASIC tokens in order to program the microcontroller. The BASIC STAMP 1C features a 256-byte EEPROM, or about 75 instructions, and runs at 4 MHz, performing 2000 instructions per second. A revision of the STAMP 1 later became the STAMP 1 rev. D. The latest version of the BASIC STAMP 1 is the STAMP 1 module shown in figure 6.3.1. The STAMP 1 is programmed via the parallel port of a personal computer. The Basic Stamp 1 is built around PIC 16C56.



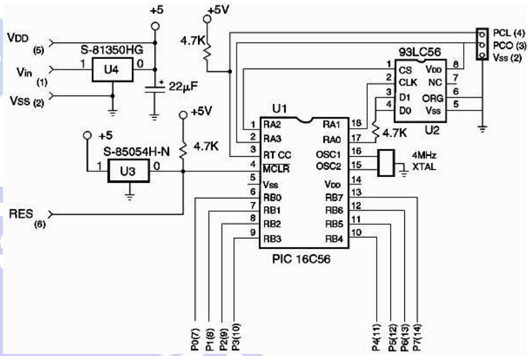
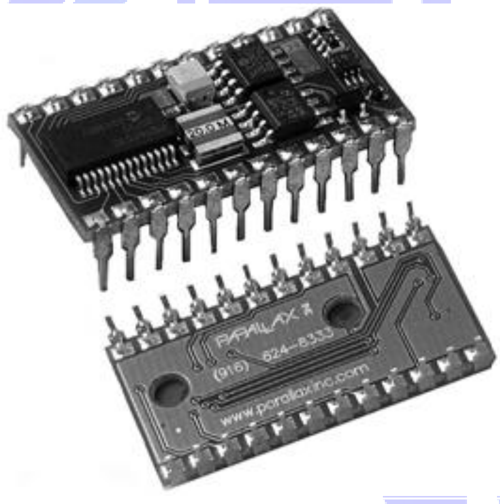


Figure 2.3.1: Basic stamp 1 housing a PIC 16C56 | Figure 2.3.2: Schematic of Basic Stamp 1

Basic Stamp 2 (BS2)

The Basic Stamp 2 is a very versatile and popular microprocessor building block that can be used to develop many different electronic sensing and control systems.



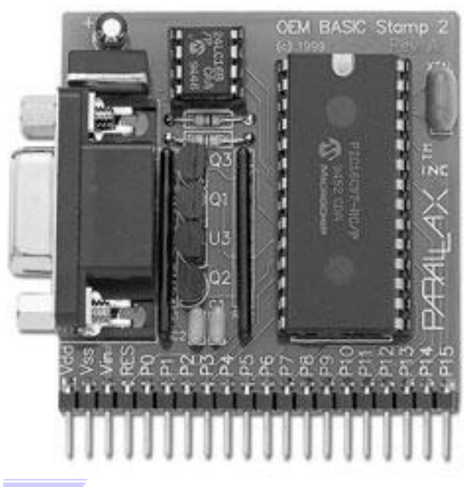


Fig. 2.3.3(a): Top and bottom view of DIP version of BS2 | Figure 2.3.3(b): SOIC version of BS2

The BASIC Stamp 2 is comprised of several components:

PIC16C57-20/SS- a Microchip 2K 8-bit microcontroller programmed with the BASIC Stamp “Interpreter” that runs the Parallax BASIC Stamp code like a mini operating system; the 24LC16B EEPROM - a Microchip 2KEEPROM (electrically erasable programmable read-only memory) with a user-modifiable read-only memory (ROM) that can be reprogrammed with BASIC Stamp code; a 20 MHz resonator to provide an accurate clock source for the BASIC Stamp; 220 Ohm resistors across the I/O pins to provide current protection for mistake wiring; an LM2936 voltage regulator which provides 50 mA for the BASIC Stamp 2.

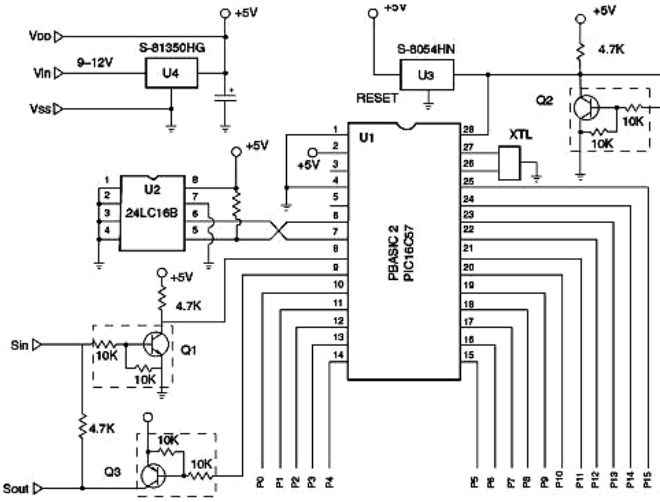


Figure 2.3.4: Basic Stamp 2 Schematic which is built around PIC 16C57

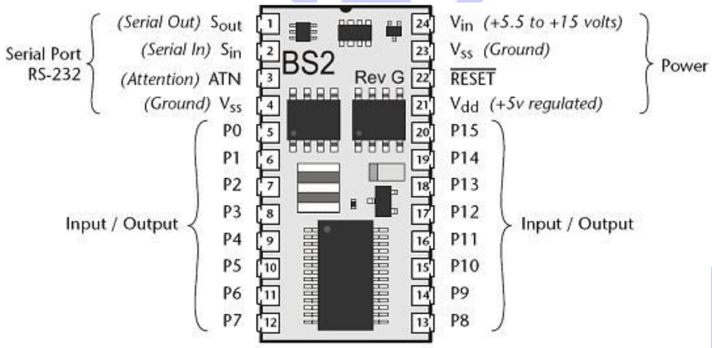


Figure 2.3.5: BS2 pin description

The BASIC STAMP 2 is available in two different types: dual inline package (DIP) or a small outline integrated circuit (SOIC) package.

Basic Stamp 2 pin descriptions

Table 2.3.1: Pin description of BS2

Pin	Name	Description
1	S_{OUT}	Serial Out: Connects to PC (personal computer) serial port RX pin (DB9 pin 2/DB25 pin 3) for programming.
2	S_{IN}	Serial In: Connects to PC serial port TX pin (DB9 pin 3/ DB25 pin 2) for programming.
3	ATN	Attention: Connects to PC serial port DTR pin (DB9 pin 4/DB25 pin 20) for programming.
4	V_{SS}	System ground: (same as pin 23) connects to PC serial port GND pin (DB9 pin 5/DB25 pin 7) for programming.
5-20	P_0-P_{15}	General purpose I/O pins: each can sink 25 mA and source 20 mA. However, the total of all pins should not exceed 50 mA (sink) and 40 mA (source) if using the internal 5 volt regulator. The total per 8 pin groups (P_0-P_7 or P_8-P_{15}) should not exceed 50 mA (sink) and 40 mA (source) using an external 5 volt regulator.
21	V_{DD}	5-volt DC input/output: if an unregulated voltage is applied to the VIN pin, then this pin will output 5 volts. If no voltage is applied to the VIN pin, then a regulated voltage between 6.5V and 5.5V should be applied to this pin.
22	RES	Reset input/output: goes low when power supply is less than approximately 6.2 volts, causing the BASIC Stamp to reset. Can be driven low to force a reset. This pin is internally pulled high and may be left disconnected if not needed. Do not drive high.
23	V_{SS}	System ground: (same as pin 4) connects to power supply's ground (GND) terminal.
24	V_{IN}	Unregulated power in: accepts 5.5 - 15 VDC (6-40 VDC on BS2-IC Rev. e, f, and g), which is then internally regulated to 5 volts. Must be left unconnected if 5 volts is applied to the VDD (+5V) pin.

Basic Stamp 2 specifications

Execution Speed: Approx. 4,000 instructions/sec

RAM Storage: 32 Bytes (6 Bytes for I/O, 26 Bytes for variables)

EEPROM (Program) Storage: 2K Bytes (approx. 500 instructions)

Voltage: 5 to15 V DC

Source/Sink Current per pin: 20 mA

Source/Sink Current per unit: 40 mA

PC Interface Serial: RS-232 (9600 baud)

Memory organization description

The Basic Stamp 2 has 32 bytes of working memory. This is not much when compared to the megabyte memories of desktop computers – but used carefully it is good enough for our purposes. It is Random Access Memory (RAM) which means that it is fast for reading and writing. (There another much larger but slower memory available on the Stamp besides RAM.) The memory is organized in 16-bit words. A word is broken down into progressively smaller parts: a byte (8-bits), a nibble (4-bits) and the single bit. In order to make the most of limited RAM, the Stamp allows memory to be addressed at the level of bits, nibbles, bytes or words. Low-order bits, nibbles and bytes are on the right hand-side and high-order on the left. This corresponds to the numbers we are already familiar with – decimal numbers. For example, in the number '2005', '2' is the high-order digit and '5' is the low-order digit. The memory organization is as shown in the figure 2.3.6.

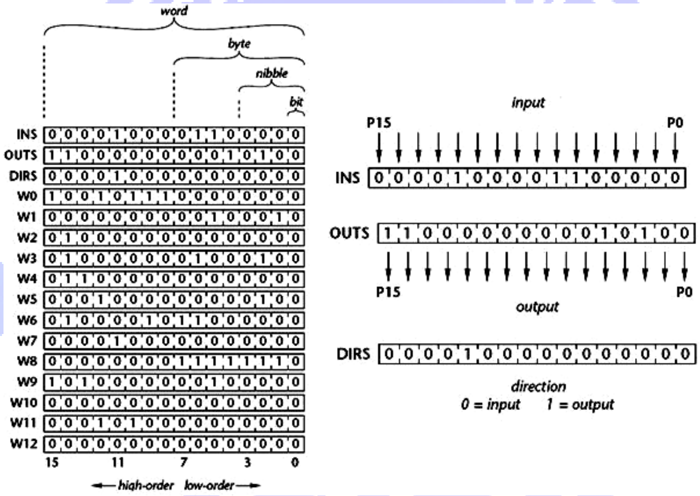


Figure 2.3.6: Memory organization of BS2

The first three words, INS, OUTS and DIRS have a special behaviour. They are memory-mapped to the input/output pins P0-P15. The first word, INS, will always contain the current state of the pins P0-P15. Reading the bit values in this word will whether a given pin is voltage HIGH or voltage LOW. This word is read-only. The second word, OUTS, contains the output values destined for the pins P0-P15. To bring an output pin HIGH, it is necessary to write a '1' to the corresponding bit in OUTS. This word is read-write. The third and final memory mapped word is DIRS and it controls the direction of data flow on the I/O pins. The pins P0-P15 can be used for either input or output at any given moment of time, and this word specifies the direction of each pin. When a bit is set to '0' the corresponding pin acts as an input. Setting the bit to '1' will make the pin an output. A pin can be changed from an input to output and back on-the-fly, under program control. The remaining 13 words, W0 to W12 are available to the programmer for storing variables. It is better to let the compiler determine the actual storage location by using the VAR command in PBasic. (e.g: "fooBar VAR Byte"). The word variable INS is unique in that it is read-only. The 16 bits of INS reflect the state of I/O pins P0 through P15. It may only be read, not written. OUTS contains the states of the 16 output latches. DIRS controls the direction (input or output) of each of the 16 I/O pins. A 0 in a particular DIRS bit makes the corresponding pin an input and a 1 makes the corresponding pin an output. So if bit 5 of DIRS is 0 and bit 6 of DIRS is 1, then I/O pin 5 (P5) is an input and I/O pin 6 (P6) is an output. A pin that is an input is at the mercy of circuitry outside the BASIC Stamp; the BASIC Stamp cannot change its state. A pin that is an output is set to the state indicated by the corresponding bit of the OUTS register. When the BASIC Stamp is powered up, or reset, all memory locations are cleared to 0, so all pins are inputs (DIRS = %0000000000000000). Also, if the PBASIC program sets all the I/O pins to outputs (DIRS = %1111111111111111), then they will initially output low, since the output latch (OUTS) is cleared to all zeros upon power-up or reset, as well.

Table 2.3.2: BS2 memory organization

BASIC STAMP 2 MEMORY MAP				
WORD NAME	BYTE NAME	NIBBLE NAME	BIT NAME	SPECIAL NOTES
INS	INL INH	NA,INB	IN0-IN7 IN8-IN15	Input pins, word, byte
OUTS	OUTL OUTH	OUTA,OUTB	OUT0-OUT7 OUT8-OUT15	Output pins, word, byte nibble and bit addressable
DIRS	DIRL DIRH	DIRA,DIRB DIRC,DIRD	DIR0-DIR7 DIR8-DIR15	I/O pin direction, control word, byte, nibble and bit addressable
W0	B0			General purpose, word, byte
	B1			nibble and bit addressable
W1	B2			General purpose, word, byte
	B3			nibble and bit addressable
W2	B4			General purpose, word, byte
	B5			nibble and bit addressable
W3	B6			General purpose, word, byte
	B7			nibble and bit addressable
W4	B8			General purpose, word, byte
	B9			nibble and bit addressable
W5	B10			General purpose, word, byte
	B11			nibble and bit addressable
W6	B12			General purpose, word, byte
	B13			nibble and bit addressable
W7	B14			General purpose, word, byte
	B15			nibble and bit addressable
W8	B16			General purpose, word, byte
	B17			nibble and bit addressable
W9	B18			General purpose, word, byte
	B19			nibble and bit addressable
W10	B20			General purpose, word, byte
	B21			nibble and bit addressable
W11	B22			General purpose, word, byte
	B23			nibble and bit addressable
W12	B24			General purpose, word, byte
	B25			nibble and bit addressable

Basic Stamp 2 and PIC

The BS2 microcontroller is built around Microchip PIC (Programmable Interface Chip) 16C57C. The comparison between Basic Stamp 2 and PIC micro controllers are as given in table 2.3.3.

Table 2.3.3: Comparison of BS2 and PIC

Parameter	Basic Stamp 2	PIC
Cost	Expensive	Cheaper
Learning curve	Easier to learn	Difficult to learn
Speed	Slower	Fast
Programming feasibility	Free software for programming (PBasic) can be downloaded from parallax website.	Requires a compiler to create software

2.4 Programming BS2 with PBasic

PBASIC stands for Parallax BASIC which is a variant of BASIC programming language. This special language has familiar BASIC instructions such as FOR..NEXT, IF..THEN and GOTO along with some useful extra instructions that are specially for input and output (I/O). Programs can be written using the BASIC Stamp programming software and downloaded on a serial port to the BASIC Stamp 2.

Figure 2.4.1 shows a screenshot of BASIC Stamp Editor v2.3.

BASIC STAMP I/O INSTRUCTIONS	
BASIC STAMP 1 (BS1)	**BASIC STAMP 2 (BS2)**
BUTTON	BUTTON
HIGH	COUNT
LOW	DTMFOUT
OUTPUT	FREQOUT
POT	HIGH
PULSIN	LOW
PULSOUT	OUTPUT
PWM	PULSIN
REVERSE	PULSOUT
SERIN	PWM
SEROUT	RCTIME
SOUND	REVERSE
TOGGLE	SERIN
	SEROUT
	SHIFTIN
	TOGGLE
	XOUT

Table 2.4.1: Set of I/O instructions in PBasic for BS1 and BS2

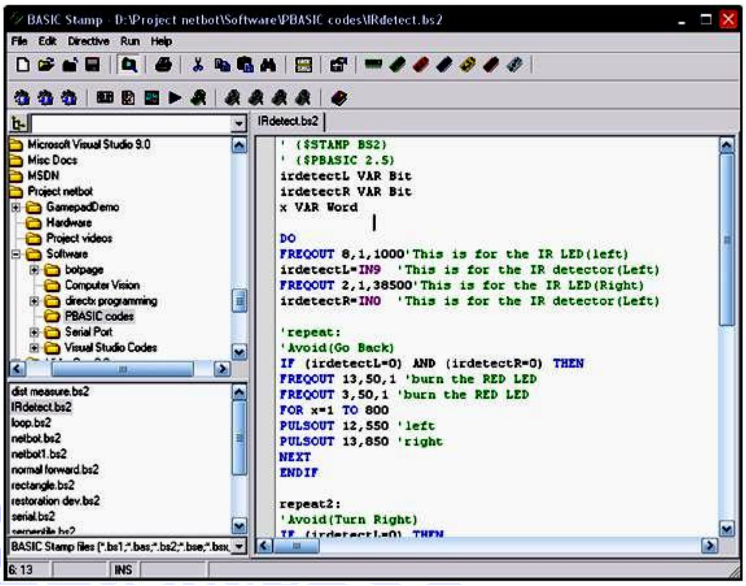


Figure 2.4.1: Screenshot of the BASIC Stamp Editor v2.3

3. Further Work Carried Out So Far

The video stream is obtained from the web camera of the robot. Using Gaussian low pass filter, the image is blurred. Naturally, the sharp edges are lost in the processed frame. The low pass filter could be applied to recorded video stream too. Our programs run only on frames stored in the video buffer. In order to process AVI files, we uncompress the AVI file, sequentially, store frames in a buffer, process the buffered frame, and sequentially display the processed frames on the monitor sequentially. Based on colour information (say red), the object is isolated after low pass filtering the video frames twice and then thresholding it with a priori colour value (RGB) and binarization.

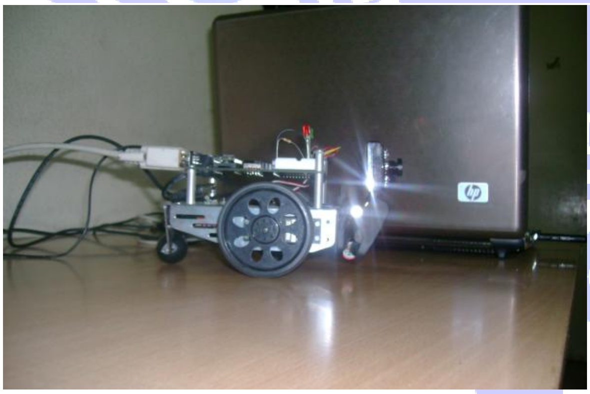


Fig. 3.1: The AMR built by the author

The isolated and binarized video frame image is highly sparsed and disconnected at various places. In such a situation, it is very difficult to fix the tracking point (say, centroid) in the image. As a feasible solution to this problem, we dilate the image using morphological dilation operator with the help of a suitable structuring element. Bigger the size of the structuring element, better the dilation so that most of the disconnected regions would get connected. The only disadvantage here is that the speed of processing the video would reduce considerably. The robot is fitted with a web camera and powered by a set of AA type batteries.

4. Conclusions

The problem of finding a feasible method for real-time tracking of a target (object) by an Autonomous Mobile Robot (AMR) using Machine Vision, Image Processing and Pattern Recognition and Tracking techniques are presented in this paper. The technique presented in this paper could be used for real time applications like autonomous homing system, space crafts and other robotic applications.

References

1. Andrew Webb, " Statistical Pattern Recognition", 2nd Edition, Wiley 2002.
2. Alexander Hornberg, "Handbook of Machine Vision" Wiley-VCH, 2006

3. Davies, E. R. "Machine Vision: Theory, Algorithms", Practicalities, Edition: 2, Academic Press, 1997.
4. Don Pearson, Image Processing, Tata McGraw Hill, U.K. 1991.
5. Giardina, C.R., and Dougherty, E.R., Morphological Methods in Image and Signal Processing, Prentice Hall Inc., 1988.
6. Groover Mikel, P., Automation, production systems and computer integrated manufacturing, Prentice-hall of India Private Limited, 1996.
7. John C. Russ, "The image processing handbook", Edition: 3, CRC Press, 1999
8. J. P Margues de Sa, " Pattern Recognition Concepts, Methods and Applications, Spinger, May 2001, Portugal.
9. Jorge L. C. Sanz, "Image technology: advances in image processing, multimedia and machine vision", Springer, 1996
10. Julius J. Tou Rafael C. Gonzales, Pattern Recognition Principles, Addison Wesley 1974.
11. Keinosuke Fukunaga, " Introduction to Statistical Pattern Recognition", Second Edition, School of Electrical Engg. Purdue University, Indiana.
12. Milan Sonka, Václav Hlaváč, Roger Boyle, "Image Processing: Analysis and Machine Vision", second edition, Thomson, 1998
13. Nello Zuec, "Understanding and Applying Machine Vision", Edition: 2, CRC Press, 2000.
14. Paul F. Whelan, Derek Molloy, "Machine vision algorithms in Java: techniques and implementation", Edition: 2, Springer, 2001.
15. Rajan E. G., "Symbolic Computing – Signal and Image Processing", Anshan, Kent, CBS, UK. 2004.
16. R. C Gonzales and R. E Woods, Digital Image Processing Addison Wesley, 2000.
17. Ramesh Jain, Rangachar Kasturi, Brian G. Schunck, "Machine Vision" McGraw-Hill, 1995.
18. Richard O. Duda, Peter E. Hart, David G. Stork, " Pattern Classification", Second edition.
19. Robert J, Schalkoff, Pattern Recognition: Statistical, structural and Neural approaches. ISBN : 0-0471-52974-5, June 1991.
20. Bray A. J. Tracking objects using image disparities. Proceedings of the 5th Alvey vision conference, Reading, 1989.
21. Chou-Ting Hsu and Ja-ling Wu, Hidden Digital Watermarks in Images, Senior Member, IEEE. IEEE Transactions on Image Processing.
22. G. Sathya, G. Prashanthi, "A Novel Approach To Extract Hidden Features of Ultrasound Imageries", July 2008, pp 66-66
23. G. Sathya, G. Prashanthi, "Set Theoretic Rajan Transform (STRT)", International Journal of Systemics, Cybernetics and Informatics, October 2007 pp 37-37
24. G. Sathya , Ruchika Goyal, A. Sirisha, C. Vandana, G. Prashanthi, "Inverse Set Theoretic Rajan Transform (ISTRT)", International Journal of Systemics, Cybernetics and Informatics, April 2008.
25. Harris C and Stennett C., "RAPID – a video rate object tracker", Proceedings of the 1st British Machine Vision Conference BMVC, pages 73-73, 1990.
26. Miljković & Babić. "Empirical Control Strategy for Learning Industrial Robot", FME Trans., 2007
27. Oh-Jin-Kwon and Rama Challappa, "Segmentation Based Image Compression", CS_TR_2863, Technical Report, March 1992, Centre for Automation Research, University of Maryland, College Park, U.S.A.
28. Rajan, E.G., and Ramprasad, V.V., Pattern-Directed Array Processing, Technical Report, TR/IP-5-91, Department of Electrical Engineering, Indian Institute of Technology, Kanpur, January, 1991.
29. Rajan, E.G., Cellular Logic Array Processing, Techniques for high throughput Image Processing Systems, SADHANA, Special Issue on Computer Vision, Volume 18, Part 2, June 1993, pp. 279-300, Indian Academy of Sciences.

30. Rajan, E.G., High-Throughput Cellular Logic Array Processing of Satellite Data for Geophysical Surveying, Paper No A.1-S.1.08. The World Space Congress, Washington, D.C., U.S.A., 28 August to 5 September 1992.

31. Rajan, E.G., Object Recognition and Tracking in a Multiple Target Environment, 1994 IEEE International Conference on Systems, Man and Cybernetics, San Antonio, Texas, U.S.A., October 2-5, 1994, pp-2390-2395.

32. Rajan, E.G., Fast Algorithm for Skeletonising 3-D Digital Images, 8th International Conference on Biomedical Engineering, December 7-10, 1994, Singapore, pp. 151-153.

33. Rajan, E.G., A Genetic Algorithmic Fixing of Attack Points in a Moving Target, 1995 IEEE International Conference on Systems, Man and Cybernetics, Vancouver, B.C., Canada, October 22-25, 1995, pp. 217-222.

34. Rajan, E.G., Neural Automata Based Object Recognition, 1995 IEEE International Conference on Systems, Man and Cybernetics, Vancouver, B.C., Canada, October 22-25, 1995, 1882-1887.

35. Rajan, E.G., The Notion of Geometric Filters and their use in Computer Vision, 1995, IEEE International Conference on Systems, Man and Cybernetics, Vancouver, B.C., Canada, October 22-25, 1995, 4250-4255.

36. Rajan, E.G., Cellular Logic Array Processing, Invited paper, World Congress for Nonlinear Analysts, organized by the international Federation of Nonlinear Analysts, Florida,\ Institute of Technology, July 10-17, 1996, Athens, Greece.

37. Rajan, E.G., Fast Algorithm for collision avoidance by autonomous mobile systems in the framework of cellular logic array processing, 13th international conference on CAD/CAM Robotics & Factories of the Future, Universidad Technologica de Pereira, Columbia, South America, Dec. 15-17, 1997.

38. Rajan, E.G., Srinivas, S.S.S., Srinivas, P.L., Mallikarjuna Rao, G., Srinivasa Rao, N., "On the notion of Generalised Rajan Transform", World multiconference on Systemics, Cybernetics and Informatics, Caracas, Venezuela, July 7-11, 1997.

39. Rajan, E.G., Prakash Rao, K., Vikram, K., Harish Babu, S., Rammurthy, D., Rajasekhar, "A Generalised Rajan Transform Based Expert System for Character Unduerstanding", World multiconference on Systemics, Cybernetics and Informatics, Caracas, Venezuela, July 7-11, 1997.

40. Rajan, E.G., Ramakrishna Rao, p., Mallikarjuna Rao, G., and Venugopal, S., "Use of Generalised Rajan Transform in Recognising Patterns due to defects in the weld", JOM—8 International Conference on the Joining of Materials, American Welding Society, JOM Institute, Helsingor, Denmark, May 1997.

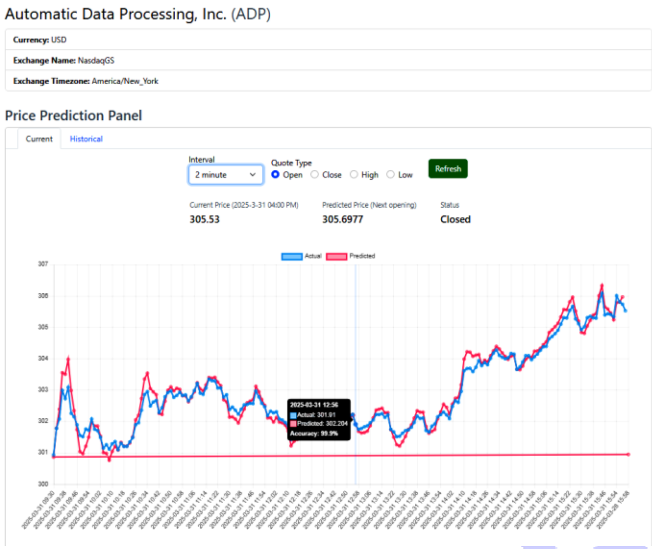
41. Rajan, E.G., Venugopal, S., "The role of RT in the machine vision based factory automation", National conference on Intelligent manufacturing systems, February 6 – 8, 1997, Coimbatore Institute of Technology.

42. Srinivas Rao, N., " Data Encryption and Decryption Using Generalised Rajan Transform", M. Tech Thesis, department of Electronics and Communication Engineering, Regional Engineering College, Warangal, 1997.

43. Venugopal K., Rajan E.G., "Pextral coding of images, National Conference organized by the Institution of Engineers, India and Annamalai University, SURGE`94, 1994.

About The Author

Sathya Govindarajan obtained B.Tech (ECE) degree from Jawaharlal Nehru Technological University, Hyderabad, India and MS degree from New York Institute of Technology, Old Westbury, New York, USA. He worked in DARPA sponsored project on continuous authentication – Key Stroke and Touch, in NYiT. He was also involved in the application development for Continuous Facial Authentication. At present, he is working as a Senior Lead in Automatic Data Processing Inc. (ADP), New Jersey, USA. His areas of interest include Data Science, Signal and Image Processing, Robotics and Automation, Advanced Communication Systems, Computer Animation and Graphics.



Sathya has developed many software products, which are of immense use in Speech and Signal Processing, 2D and 3D Image Processing, Data Analytics, Automation, Machine Vision to name a few. Stock Price Prediction Support System (SPPSS) is a cloud based software system developed by Sathya Govindarajan in JAVA environment. The AI based prediction algorithm developed by him ensures a prediction accuracy ranging from 99.8% to 100 %. He has used concepts and tools from linear dynamical system, probability theory, detection and estimation theory in the development. A sample display of the predicted time series is shown on the left. This technique has potential applications in business, healthcare, defence, agriculture and many other domains where data analysis and prediction are used extensively.

♦♦♦

Mr. Sathya Govindarajan has been rcognized as a ‘Distinguished Scientist’ by the steering committee members of the international conference on science, spirituality for global peace and harmony (IAPIC 2025). This citation was awarded to him in absentia and the PDF version sent to him. The organizing and steering committee members congratulate him for his innovative contribution in the field of ‘Machine Vision’.