System Requirements Specification

CS-308 Project

Self Balancing Bot

Group-14
Vinayak Gagrani (09005035) TL
Nitish Jhawar (09005032)
Tarique Aziz (09005036)
Vaibhav Krishan (09005042)

Submitted in partial fulfillment of the requirements of CS-308 Embedded Systems

Contents

Chapter 1

Introduction

The purpose of this document is to present a basic idea of implementation of Self Balancing Bot as the course project for CS 308. We have tried to cover basic idea of self balancing, hardware requirements, physical constraints and software & algorithm behind the same. We have also tried to include the response to various external situations and fall back strategy for extreme situations.

1.1 Definitions, Acronyms and Abbreviations

Terms	Description
Bot / Robot	The electromechanical machine that will operate according to our program.
Software Requirement Specifications	The document all the functionalities of our robot and the environment along with all the constraints under which it will work.
IDE	Integrated Development Environment.
AVR Studios	IDE for C programming that will run the bot.
Accelerometer	A device to measure of acceleration of bot in horizontal and vertical plane.
Gyroscope	A device for measuring or maintaining orientation based on principle of angular momentum.
PID Controller	A proportionalintegralderivative controller (PID controller) is a generic control loop feedback mechanism (controller) widely used in industrial control systems a PID is the most commonly used feedback controller.

1.2 References

1. `http://www.geology.smu.edu/~dpa-www/robo/nbot/` last viewed 10th Feb 2012
2. `http://www.youtube.com/watch?v=iDAUMCVHV8g` last viewed 9th Feb 2012
3. `http://www.youtube.com/watch?v=oxsCyere8hk` last viewed 9th Feb 2012
4. `http://en.wikipedia.org/wiki/PID_controller` last viewed 1st March 2012

Chapter 2

Overall Description

2.1 Aim of Project

] The aim of the project is to operate a two wheel self balancing bot. Project aims in steps to achieve the following :-

1. Balance the bot in a steady position
2. Move the bot forward and backward
3. Make the bot to make halts while moving in a linear path and resume movement again
4. Make bot take soft turns
5. Merge all the features to then make bot move in any plane
6. Remotely control the bot using a camera and zigbee

Once balanced any other team or project can then directly include these modules to later extend it to perform various other tasks like surveillance.

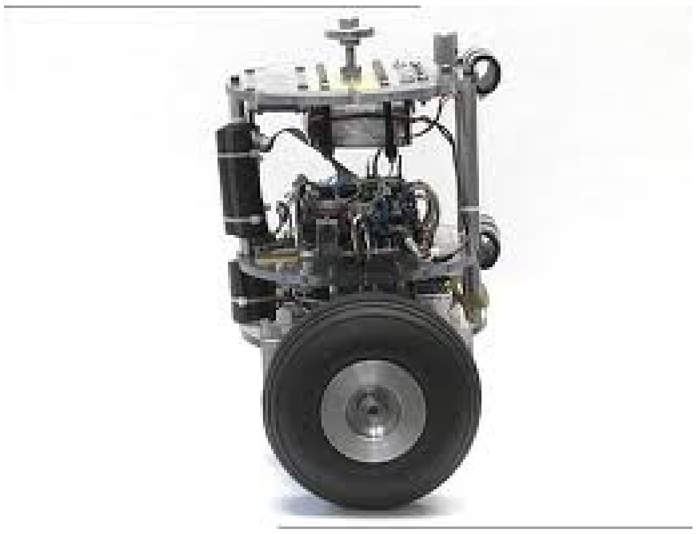

Figure 2.1: Balance Bot

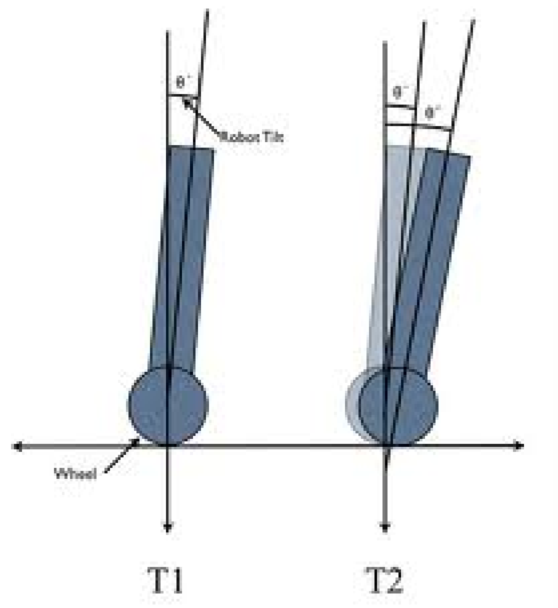

Figure 2.2: The vertical changes in parameters

2.2 System Environment

The bot will be moving in any room as environment. Preferably a planar room or planar inclinations are suggested. Since it does not have a caster wheel after the implementation so it does not have issues of getting stuck in small grooves and holes. The bot can be equipped with a camera and a zigbee(wireless device) to transfer the video feed at regular intervals as well as controlling the bot from a remote system or location.

2.3 Product Functions

Bot can be used to perform any task which firebird can perform. Rather the bot has indigenous advantage over firebird as it does not have a caster wheel.

1. The bot can be equipped with a camera and a zigbee(wireless device) to transfer the video feed at regular intervals as well as controlling the bot from a remote system or location.
2. The end product is a robust and efficient implementation of a balance bot which moves based on instructions from external and may be remotely located user.
3. We intent to develop a module which can have all possible movement commands exported for use by any other module.

2.4 Distinguished Modules of the Project

these features can be implemented as four different modules. Each module caters to one specific function and thus allows for more precise implementation and use. This also allows for easier improvement or modification later on.

1. Balance at static position

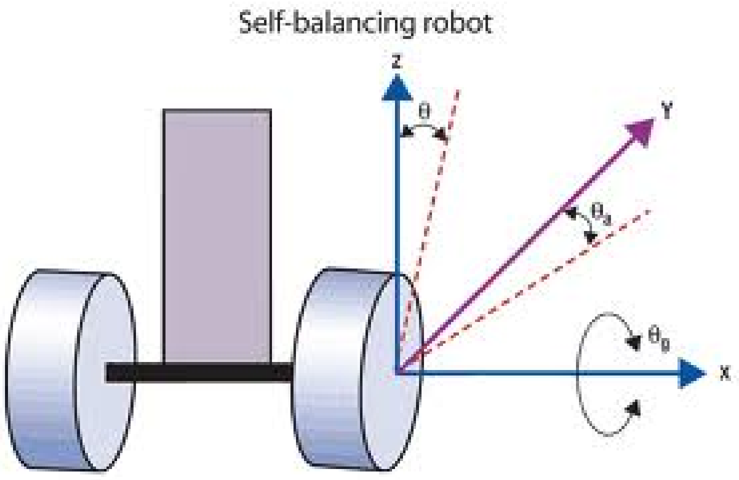

Figure 2.3: Changes during motion

2. Movement linear direction
3. Rotation at static position
4. Rotation while movement

2.5 Implementation

2.5.1 Algorithm

The PID controller calculation (algorithm) involves three separate constant parameters, and is accordingly sometimes called three-term control: the proportional, the integral and derivative values, denoted P, I, and D. Heuristically, these values can be interpreted in terms of time: P depends on the present error, I on the accumulation of past errors, and D is a prediction of future errors, based on current rate of change.[1] The weighted sum of these three actions is used to adjust the process via a control element such as the position of a control valve, or the power supplied to a heating element.

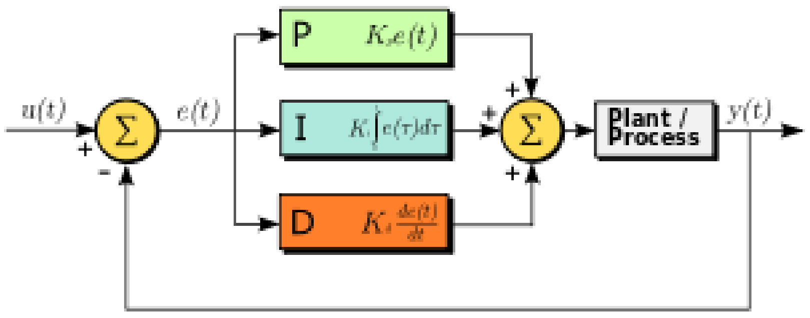

Figure 2.4: A block diagram of PID controller

Stability

If the PID controller parameters (the gains of the proportional, integral and derivative terms) are chosen incorrectly, the controlled process input can be unstable, i.e., its output diverges, with or without oscillation, and is limited only by saturation or mechanical breakage. Instability is caused by excess gain, particularly in the presence of significant lag.

Generally, stabilization of response is required and the process must not oscillate for any combination of process conditions and setpoints, though sometimes marginal stability (bounded oscillation) is acceptable or desired.

Optimum behavior

The optimum behavior on a process change or setpoint change varies depending on the application. Two basic requirements are regulation (disturbance rejection staying at a given setpoint) and command tracking (implementing setpoint changes) these refer to how well the controlled variable tracks the desired value. Specific criteria for command tracking include rise time and settling time. Some processes must not allow an overshoot of the process variable beyond the setpoint if, for example, this would be unsafe. Other processes must minimize the energy expended in reaching a new setpoint.

Limitation

While PID controllers are applicable to many control problems, and often perform satisfactorily without any improvements or even tuning, they can perform poorly in some applications, and do not in general provide optimal control. The fundamental difficulty with PID control is that it is a feedback system, with constant parameters, and no direct knowledge of the process, and thus overall performance is reactive and a compromise while PID control is the best controller with no model of the process,[2] better performance can be obtained by incorporating a model of the process.

2.5.2 Test Case

We hope to achieve following targets with respect to the functions.

1. Balance at static position : for 1 min
2. Movement linear direction : for 3 meter
3. Rotation at static position
4. Rotation while movement : soft turns

We intend to achieve the bot to move in a curved square path of side 3 m. We will try to resolve the maximum safety speed using hit and trial. The speed and acceleration will be displayed on LCD.

2.6 Basis for future work

There is a lot of prospects for further work on the bot. Since these bots can be very small , they can be used for tasks that need to be performed in compact and limited space. This will open up the area of using balance bots in projects for all the students. Various projects in firebird can be extended by other groups later to work on balance bot also.

Chapter 3

Requirements

3.1 Hardware

1. Balance Bot - The basic bot should not be costlier than firebird. In fact It can be much cheaper as many other functionalities are removed. It needs a microprocessor, sensors - proximity, buzzers.

2. Beside the general hardware of firebird, We need an accelerometer, gyroscope(optional for better precision).

3. For surveillance it needs a small camera which can take high resolution images (vector images if possible). Series of images can then be used to form a rough video clip.

4. For transfer of data(images,clips) it needs a high bandwidth wireless device (ZigBee XBee 802.15.4 OEM RF module 2.4 GHz)

5. Based on the functionality which the user will use this module and bot for, they can add other resources over and above this module to extend the balance bot functionalities.

The inputs are going to be:

1. The rate gyro: it will tell if the robot is rotating. If it's 0, then the robot is not rotating, that is: it's perfectly still, or going in one direction with some fixed speed keeping balanced. It doesn't say anything about where is up and where is down.

2. The accelerometer X axis tells if the robot is accelerating forward or backward. It can possibly be used to understand which way are we running or as a dumping factor when the robot change direction (need some actual testing). It will be influenced by the gravity when not parallel to ground.

3. The accelerometer Y axis should be able to tell where is up and where is down, since it will be influenced mainly by gravity. It will be influenced by the robot acceleration when not perpendicular to ground, but i should be able to use X acceleration to correct the reading.

4. The wheels encoders: they will tell how fast the wheels are spinning so it's the velocity of the robot.

3.2 Software

1. AVR Studio 4
2. AVR Bootloader
3. Esterel related software if needed.

Chapter 4

Risk Management

4.1 Constraints

The bot can be constrained over the speed and turning radius. Since the bot has to be balanced at all times, there will be constraints over acceleration of bot. The bot is also constrained with the weight it can hold and the height.

Fall Back We hope to provide some safety parameters about speed & acceleration based on height and weight of robot. Within the limits the bot is expected to work efficiently.

4.2 Bot Size

Since the current bot is not very small, it cannot serve as such many features of self balancing bot. The PID algorithm needs to be altered in case the dimensions of the bot change.

Fall Back Once the balancing module is complete then the hardware can be improved by removing unnecessary hardware and make a very compact design. The modules can be made to derive information about the robot(dimensions and locations of sensors) from a configuration file.

4.3 Surveillance using camera

Camera feed from the Robot: The camera feed from the robot will require a good bandwidth to transmit the raw video feed to the coordinator in order to allow user to view this feed. The ZigBee wireless communication module and the internal bus of the robot must be able to support such high bandwidth to allow this video feed.

Sample Calculation: A standard Mobile Camera with a resolution setting of 320 X 240 with an 8-bit depth will take a single picture for 32kB (gif - uncompressed raw format). For a video feed, we will require a frame speed of 13fps minimum which implies a transmission rate of 3.25 Mbps for the video alone. Also, a video clip from the same camera is also recorded but is in compressed format (AVI) @ 1.125 Mbps.

4.4 Environmental

The movement and balancing of bot will vary greatly from planar to inclined terrains. Smooth and rugged surface are also expected to give sudden changes in sensors.

Fall Back We will only test the bot for a planar and smooth surface. If time permits we will consider accounting for inclined but smooth surface.