# Rode Robots

A showcase of my exciting journey in robotics as a hobby... Each item inside consists of a story about how exactly the robots accomplished the task assigned to them. These are actual robotics competitions in which I contributed as a participant or as an organizer, or simply as a helper.

I hope it fascinates you!

**RODE ROBOTS** is a collection of some of my best works in **robotics as a hobby** during my undergraduate college (**IIT**) days. It contains my experiences from our **annual technology festivals**. It is a result of my **addiction** to robots, and is meant to give you an insight into my **passion** for robotics.

My robots accomplished the following tasks.

1. **Mazor** found its way out of a maze.
2. **Lizardus** climbed a wall.
3. **Trajector** calculated derivatives on its way.
4. **Titan** busted marine balloons.
5. **Thunder** glided through winds.
6. **Axotic** snatched balls of opponent's robots.
7. **megaMouse** ran a rat's race.

In the following pages, each robot is described in terms of **Construction** (material, dimensions and process) and **Functioning and Challenges** (how the robot accomplished its task).

# Introduction

# 1. Mazor

**Mazor** was a **semi-autonomous** robot that used **Dijkstra's algorithm** to find and travel the **least weighing path** from a source to a destination on an **8x8 colour coded chessboard** (see figure) by processing its live image using **MATLAB**, all in about 15 seconds.

## Construction

Mazor received instructions from a **computer** through its **parallel port. IC**s on Mazor interpreted the computer's signal to control two **stepper motors** that powered its left and right wheels. The wheels had a diameter of about 5 cm. The bot was about 11 cm long and 9 cm wide. It **turned in-place** and had a maximum speed of 0.4 m/s. The chessboard arena on which it travelled had 64 square cells each coloured either cyan, magenta, yellow or green. Each colour had a **weightage** that was disclosed only during the **run time**.

## Functioning and Challenges

The aim of the bot was to travel on that path whose cell weightages added to give the **least total weight**. A camera, placed (by the organizers) at a height of about 10 feet above the arena was meant to take **live** image of the **robot's position** on it. Initially an image of the arena and the weightages of the four colours were provided to the computer, which processed them in MATLAB. This was the most **challenging** part where a detailed understanding of **image processing** was required. The computer extracted the coordinates and weightages of every cell of the arena and found the least weighing path using Dijkstra's algorithm. It then gave instructions to the bot through the parallel port. The position of the bot was constantly **monitored** by the camera above the arena to check any **deviation** from the designated path. If such a deviation existed, **modifying** the path of the bot in the run time was a challenging task.

# 2. Lizardus

**Lizardus** was a manual robot that was meant to **cross a raised platform,** 1.5 foot tall and 1 foot thick and then **travel** 3 feet, fast enough to complete this assignment in **20 seconds**.

## Construction

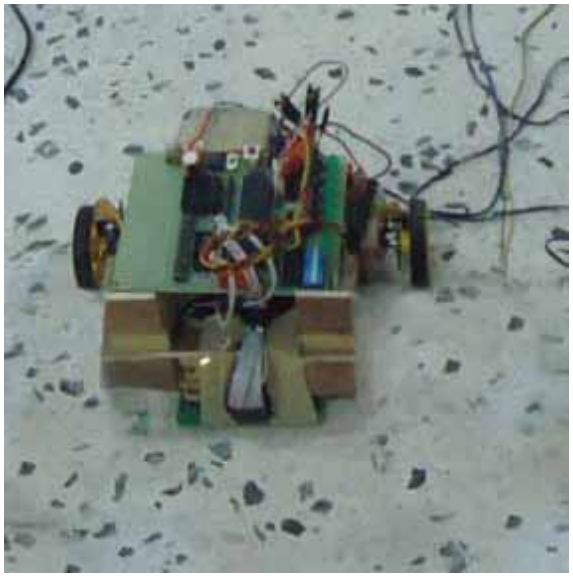
Lizardus was simple in construction. It had **three** component limbs L-shaped, A-shaped and I-shaped (or simply **L-limb, I-limb** and **A-limb** respectively), made from **wooden bars**. It had four **geared wheels** that operated through **one motor** (as there were no turnings on the way). These wheels were attached to the A-shaped limb. Two other **motors were used at pivots**, one between A-limb and I-limb, and one between I-limb and L-limb. All the motors were manually controlled by a **remote controller**.

## Functioning and Challenges

Lizardus would **initially** be in such a shape (see the figure) that it would **accommodate in a cube of volume 1 cubic foot** while its limbs would be **folded as close to each other as possible** and its wheels would rest on the ground just besides the wall. The L-limb (which was farthest from the wheels) would then be **unfolded by the pivot motors** and **raised** to rest on the top of the wall. The A-limb (with wheels) would now be **lifted up** by the pivot motors. Thus Lizardus would be back in the folded state, but this time **resting on the L-limb** instead of wheels. Finally the robot would unfold its arms on the **other side of the wall** to rest on its wheels and **travel** the specified distance of 3 feet. The major **challenge** was to **distribute the mass** of the robot in such a way that it **does not topple** while climbing up to or coming down from the raised platform. The **gears at the pivots** were to be adjusted such that there was a **balance** between the **torque obtained** from the motors and the **time taken** to turn an angle of 90 degrees.

# 3. Trajector

**Trajector** was a **semi-autonomous** robot that successfully processed an image of a 0.2 meter **continuous curve** in **MATLAB** and **traced** a 10x scaled, 2 meter **congruent trajectory** on the ground.



## Construction

Trajector had two wheels powered by **stepper motors.** A **pen-tip** was attached to it in such a way that the points at which it and the wheels touched the ground formed an **equilateral triangle**, the pen being at the rear tip of the bot. The bot was controlled by a **computer** through its parallel port. The bot contained **IC**s that interpreted the signal and sent it to the stepper motors. An image of a **0.2 meter curve** (that was to be traced) was provided, without its mathematical description, by the organizers.

## Functioning and Challenges

The image of the curve was processed in MATLAB. Robot's **initial** position corresponded to one of the **ends** of the curve in the image. An **algorithm was formulated** whose basic idea was as follows. At **every** point on the image of the curve, a **circle** of radius 1 mm was **superimposed**. The **angular coordinates** of the **points of intersection** of this circle and the curve were used to calculate the rate at which the slope of the tangent to the curve changed (the second derivative) with respect to the circle's centre. These **second derivatives** of the curve (whose mathematical description was **not disclosed** by the organizers) at all its points were collected and analysed to find the **angles** by which the two wheels of the robot should **rotate** about their respective axes. Using basic **trigonometry** and geometry, the motion of the wheels was so decided that the pen attached to it would trace the desired locus. The computer sent this data to the robot's ICs, which then sent it to the wheels. However, there was no mechanism to monitor the bot's position and check whether it **deviated** from its trajectory due to slipping of wheels on the ground.

# 4. Titan

**Titan** was a **manually** operated **marine** robot that ruptured 6 **underwater balloons** (2 more than its opponent) at **different depths**, and **swam** 180 cm, all in **39 seconds**.

## Construction

Titan consisted of three **DC motor turbines**: a **3 volt fan** on its front with a horizontal plane of rotation, and two **6 volt fans** on the sides. All motors were **sealed water-proof** and attached to a **cross-shaped wooden frame** with **water-resistant adhesive**. A thermocol block was placed at its **centre of mass** to provide **buoyancy**. The bot measured 18cm (width) x 15cm (length) x 6cm (thickness). A needle was placed at the head to burst the balloons on its way. All the motors were controlled by a **remote controller**.

## Functioning and Challenges

Using the three turbines the robot could move and turn in **any** direction. The two turbines on the sides let it **transverse** forward or backward and **turn** sidewards when one of the two fans was in operation, thus providing **two degrees of freedom**. The horizontal fan on its front was operated for **short times** to **roll** the bot about an axis parallel to the line joining the other two motors, thus provided the remaining **four degrees of freedom**. (The figure is an image taken from above the water's surface.) Thus the robot could move in any desired direction so that the targeted balloon was **straight ahead** of it. A challenge here was to **learn** to smoothly and efficiently operate the robot and rupture a balloon **before** the opponent did, and then **swim** to a destination. Manoeuvring Titan was a **feat of dexterity**.

# 5. Thunder

**Thunder,** a very light **glider aircraft** with small **electric motors,** flew **300 feet** within **30 seconds** of launch.

## Construction

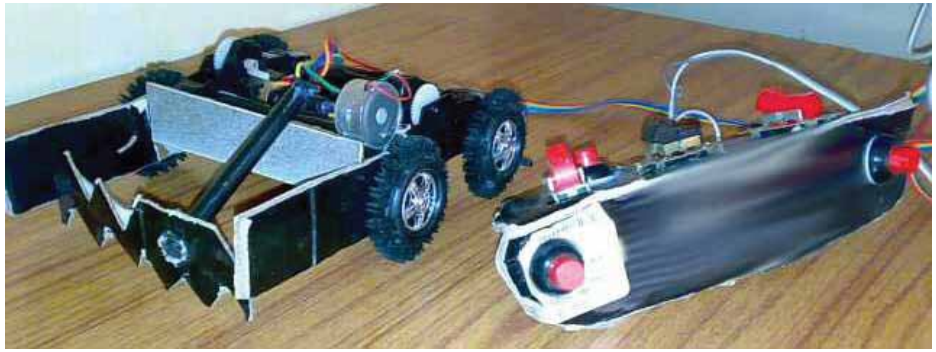
Thunder was constructed right from scratch using **solid foam**. Its **wingspan** was around **150 cm**. Its body was a thin bamboo-wood bar. **Vertical** and **horizontal stabilizers** were fitted to its tail. The construction demanded huge patience to attain perfect **angle of attack** of the aircraft's wings. A **3 volt battery** and two **DC motors** were mounted on the wings. The motors were **parallely** connected to the battery and **manually** controlled by a **transmitter-receiver**.

## Functioning and Challenges

The glider was **hand launched** and then motors were started to attain **full speed**. (The figure shows a snap edited from a video.) An **angle of elevation** of about **30 degrees** from the ground was perfect for launching. Both motors operated simultaneously, thus maintaining a **constant direction** throughout the flight. As the range of good signal strength of the transmitter-receiver was only about **500 feet**, the remote controller was operated from about 250 feet from the launch pad, in the direction of flight. Care had to be taken to ensure that the aircraft did not soar **too high** or **too far** to escape from the transmitter's network. After the competition ended, we were tempted to place a small camera on the glider and see how it would feel if we were to fly on it!

# 6. Axotic

**Axotic** was a manual robot that travelled in a maze, collect table-tennis balls from its opponents' territory and fled.



## Construction

Axotic had four **geared wheels**, two on each side, and two motors, one on each side, to power the wheels. It had an **arm** that collected table-tennis balls into a compartment at its front. A motor provided **torque** to raise and lower the arm. A **remote controller** was **ergonomically** constructed for the bot. On its **front panel** were two thumb-operated joysticks that controlled the bot's motion and direction, and a **push-button** that raised its arm. The **top panel** had two push-buttons, accessible by **index fingers**, that provided extra **booster voltage** to the bot's wheels to flee from opponent's territory.

## Functioning and Challenges

This competition was a **time challenge** among **four** contestants. All the four robots had to start **simultaneously** from four corners of a **square maze arena**. The complexities of the maze at each corner were similar. One table-tennis ball was placed in each of the four contestants' quarter squares. The challenge was to collect the **maximum** number of opponents' balls and then flee back to home quarter in **minimum** time. The overall **points** of a robot were calculated based on two criteria: how many balls (atleast one required) it collected; and how quickly it returned back to home quarter. Thus this competition tested how **efficiently** one controlled the robot. This is where our **ergonomic** design of the remote controller played a vital role. In addition, our team employed a **trick** of applying 'booster-voltage' to the wheels' motors while fleeing back to the home territory, while our opponents' robots generally travelled at constant speeds.

# 7. megaMouse

**megaMouse** was a **fully autonomous** robot mouse constructed for the popular event called ***micromouse.***

## Construction

megaMouse's **dimensions** were 13 cm (length) x 10 cm (width) x 9 cm (height) and weighed about **700 grams**. It used **stepper motors** and four **Li-Ion 2200 mAh battries.** It had a maximum speed of **0.5 m/s** and turned **in-place**. An **ATmega16 processor** was used and an **SPI based in-circuit programmer** was made. Better performing equipment would have drastically improved the mouse's speed, but I decided to experiment with simpler equipment before using high-end hardware. (In the near future I am planning to construct a micromouse with dsPIC33F processor, Faulhaber 1724006SR PMDC motors with IE-512 encoder and Li-Poly battries.)

## Functioning and Challenges

A **16x16 maze** is provided and an autonomous robot mouse is expected to first find the **shortest path** to solve it and then travell on that path with its **highest speed**. We used three **LED**s to shine light on the maze walls, and **sensors** to measure the reflection which was processed through an **AD converter** to derive the digital signal. A challenge here was to take **ambient light** into consideration, and subtract it from LEDs' light. **Flood fill algorithm** was used to solve the maze. Code was written in **C** language. A **MakeFile** was generated using MakeFile Generator. The C code was then **compiled to a hex file** using **AVR** (for which **GCC** has support). MakeFile **simplifies** this compilation. The hex file was finally **burnt** onto the ATmega16 microcontroller. **Debugging** the C code for runtime errors was quite challenging and gave a good learning experience.