Course Project Documentation

CS308 Project

Remote Controlled Robotic Arm with Parallel Simulation

Group 12

B.Sravan Kumar, 09005046
S.Chandra Mouli, 09005045
B.Kiran Maruthi, 09005047
Sonika Malloth, 09005054
D.Venkat Ramana, 09005059

Contents

1 Introduction

1.1 Motivation

Often we see many applications of mechanical devices in our life. Some are directly operated by humans while some are remotely operated, by a remote controller. The motivation for our project is to design a system, where a robotic arm is controlled by a remote, which is very intuitive to use.

Even a user who is operating it for the first time should have no difficulties. He should be able to easily maneuver the device. We came up with a mechanical arm, modelled very similar to the actual robotic arm, so that it is very intuitive to operate it.

1.2 Modules:

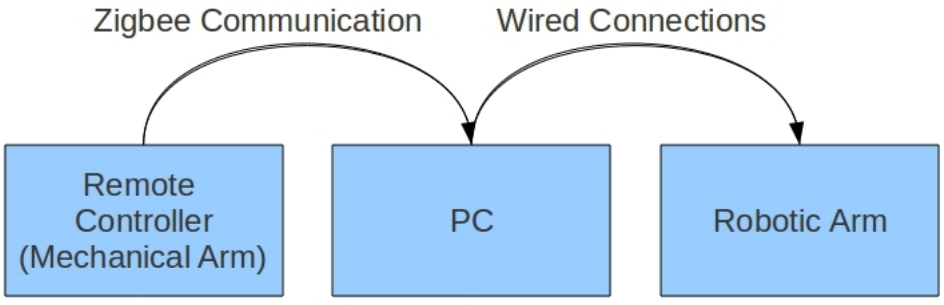
Our project essentially consists of two main segments:

- Simulation of the mechnical arm's movement in blender - a free open source 3D content creation suite.
- Replicating movement of the remote controller, ie the mechanical arm in the robotic arm.

1.3 Problem Statement and project description

The mechanical arm allows a user to control a robotic arm from a remote place. The remote control is a mechanical arm which controls the movements of the robotic arm and its jaws. Simultaneously, a simulation of the mechanical arm's movement is shown in blender, in the computer. The various degrees of freedom are clearly depicted in the simulation, which helps better understand the movements of both the arms. Analysis can be performed on the data being used for the simulation for any improvements.

The user makes the desired movements in the mechanical arm which are replicated in the robotic arm, thus allowing fexible movements of the arm which can be at a considerable distance from the user.



2 Requirements

The following are the respective hardware and software used in our project:

2.1 Hardware Requirements

- Specialized Mechanical Arm mounted on Spark 5 : for recording the required movements
- Xbee : for wireless communication between spark 5 (mechanical arm) and PC
- Potentiometers (5) : used in the mechanical arm for recording angles
- Robotic Arm : The target for replicating movements

2.2 Software Requirements

- Embedded C : for configuring Spark 5
- Blender 2.62 : for the simulation
- Python 3.2 with PySerial : for blender simulation and communication with robotic arm

3 Implementation

3.1 Functionality

- Simulation : The mechanical arm is moved to the desired position, and the five potentiometers give angles which represent the movement of each of its components. In the simulation, new position of the arm is calculated from the angles supplied by the potentiometers, and the simulation reflects this change. Thus, every movement made in the mechanical arm is simulated in blender in the computer.

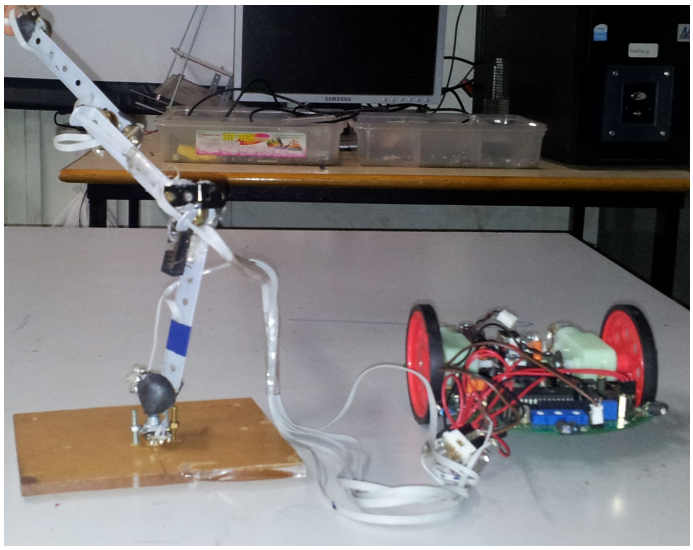
- Robotic Arm : Simultaneous to the simulation, the angles sent by the potentiometers are used to calculate the new position of the robotic arm as well. The new postions and speeds of the servometers in the robotic arm are communicated to the robotic arm. The movement is then made according to the speed of the motors sent.

4 Testing Strategy

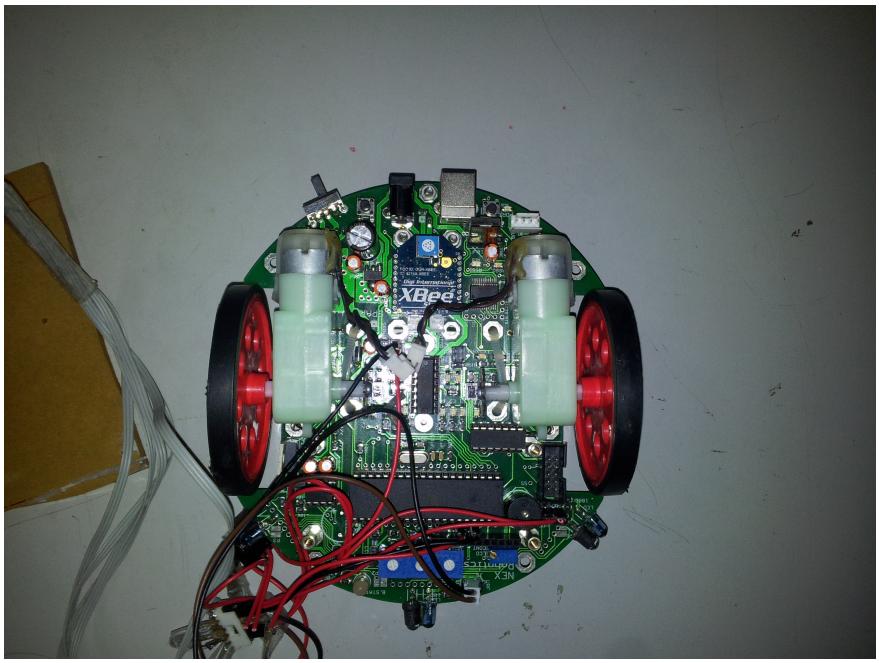
GOAL: Picking up an object at a distance and placing it at the target area.

Firstly, we shall see the major components:

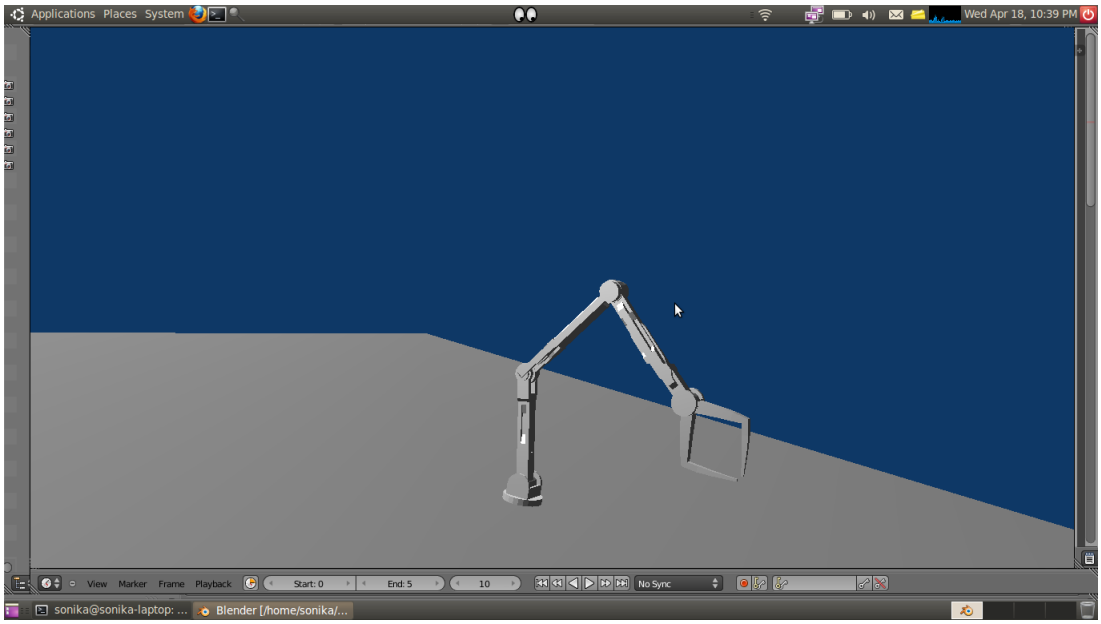
a) Mechanical arm:



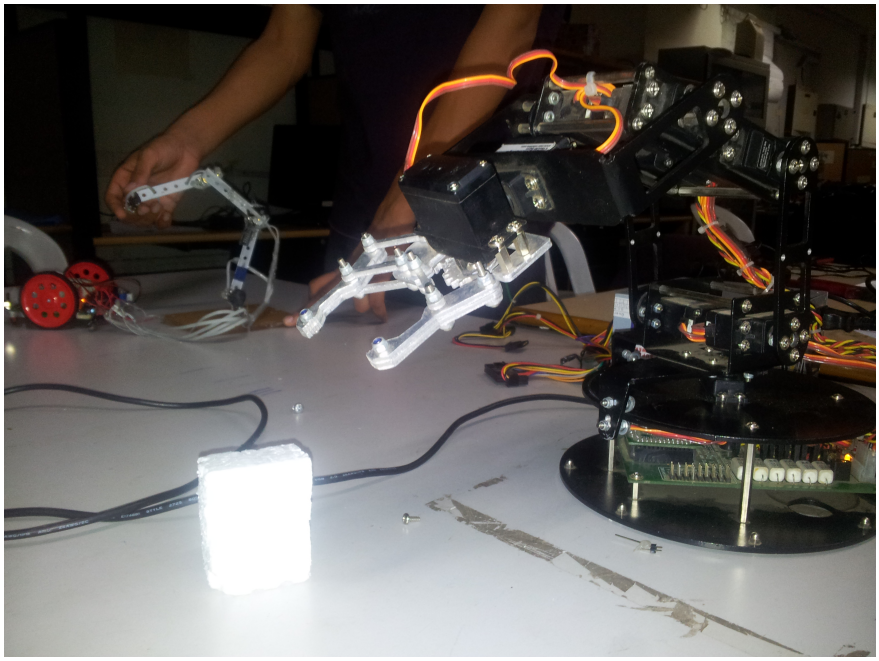
b) Spark 5 bot:



c) Simulation:

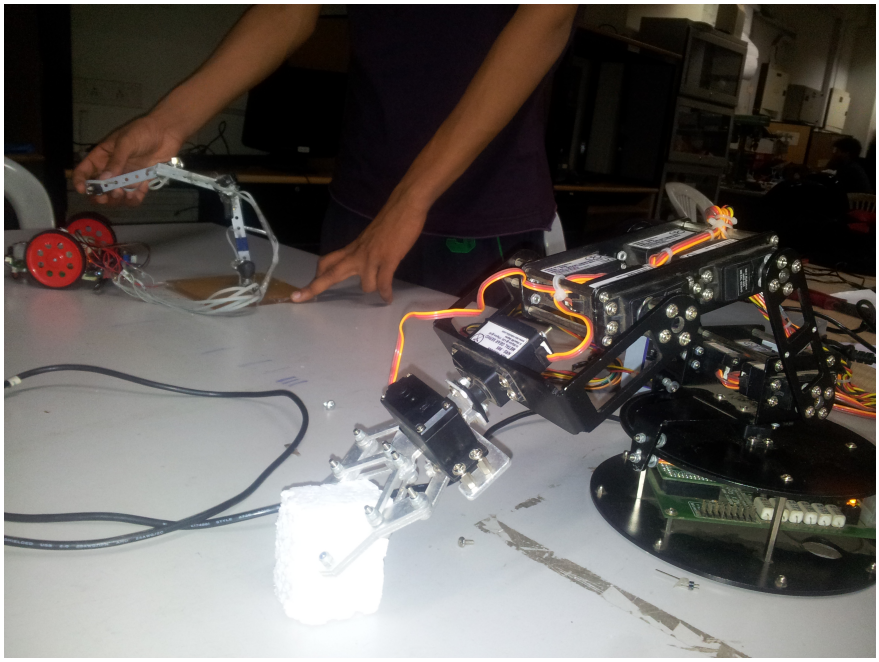


d) Robotic Arm:

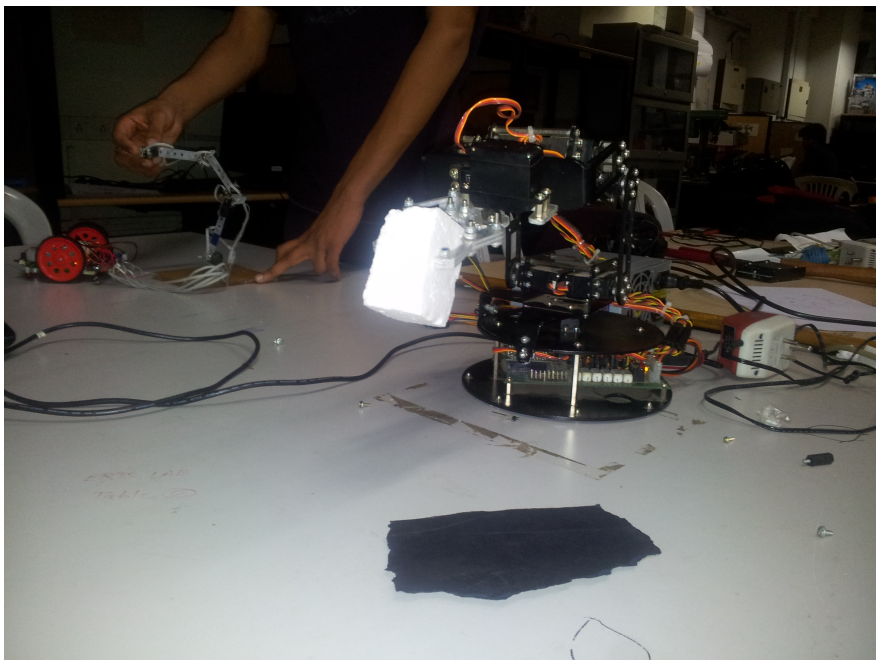


4.1 Demo

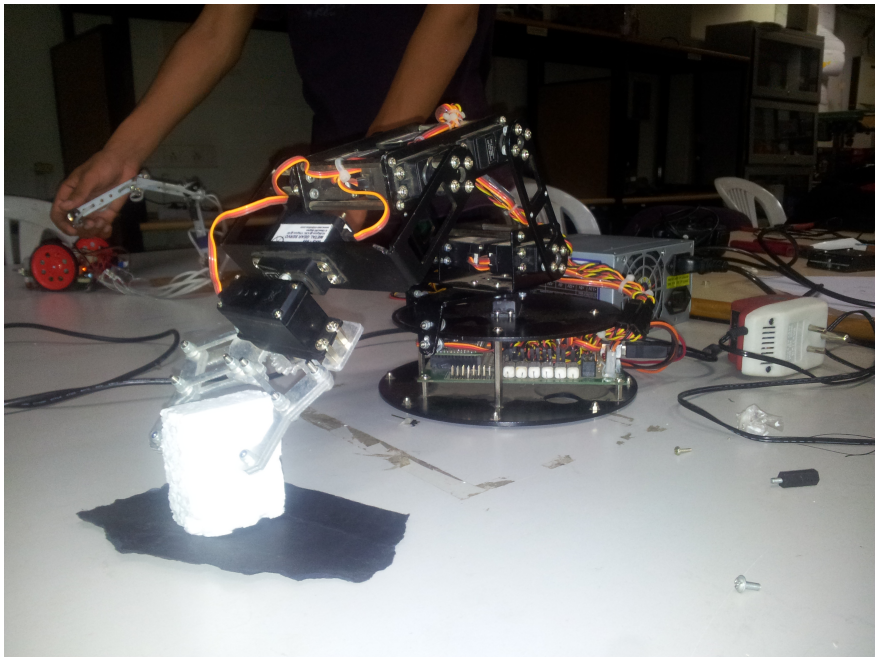
Moving a piece of thermocol from one place to another:
a) Initial position:The arm makes contact with the piece of thermocol and holds it in its jaws.



b) Intermediate postition: The motion has started, and the piece is moved from the starting postion. It is currently in an intermediate position, the goal is to place it on the black piece of paper:



c) Final position: The arm is then moved above the target area and the block is gently placed over the target area. The movement of jaws is done by the rotation of the topmost potentiometer on the mechanical arm:



5 Discussion of the system

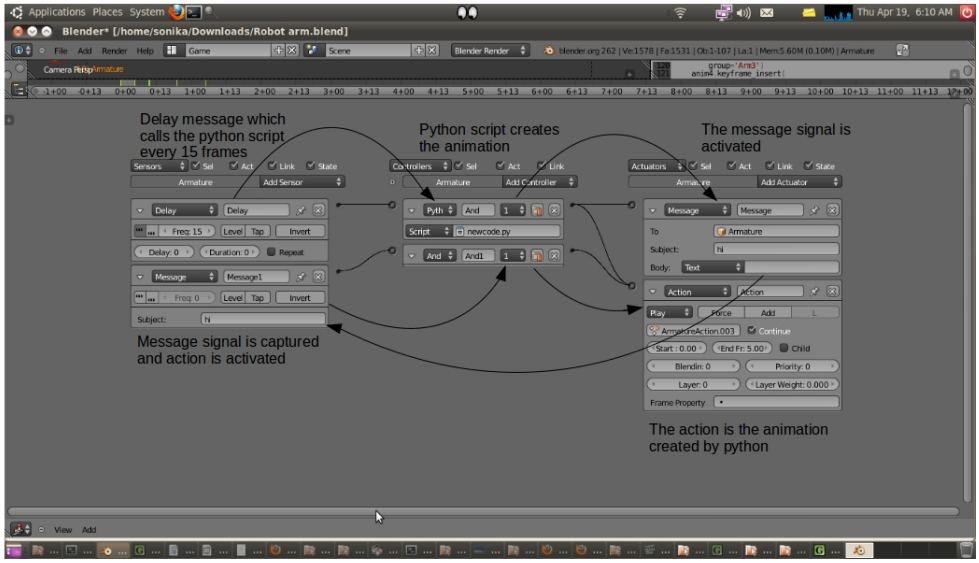
5.1 Modules working according to initial plan

- Mechanical Arm: Successfully constructed an intuitive model of the mechanical arm.
- Potentiometers: The potentiometers have been installed successfully in the mechanical arm to give the angles to the PC, according to the movement of the arm.
- Processing: All the processing has been done with python scripts on linux platform. The values given by the potentiometers are not linear. They are calibrated with precision to produce the angles to be given to the blender.
- Python: Python was used for communication with blender for the simulation as well as to communicate with the robotic arm to send the final positions.
- Blender software: Python script consists of data for the final position of the Robotic arm, along with the speeds of respective servomotors with which the movement is to be carried.
- Communication with the Robotic Arm: We decoded the format of communication string to the robotic arm. It consists of a '#' followed by the motor number which we want to control along with its absolute angle and speed. Four stars marks the end of the signal. This can be ported to any programming language on any operating system to control the robotic arm. For example - #1AB**** will signal motor 1 to rotate with speed of 66 and set the angle to 65(A) absolute.
- Robotic Arm: The robotic arm successfully replicates the movements of the mechanical arm, which was our final goal. The arm moves smoothly to the new positions. The movements of the jaws were also smoothly implemented.

5.2 Extra modules added

- Simulation : Simulation of the movements of mechanical arm was implemented in blender. This is a very big addition to our project. The values from potentiometers were used for the simulation. We have written a game logic in blender to run the simulation in real-time. First a delay sensor is started which will run with a delay of 15 frames. For every 15 frames the python script which creates the animation is called for. This script activates a message actuator which inturn sends a message to a message sensor after creating the animation. Message sensor sends a signal to the AND logic which activates the action actuator with action as the created animation.

Blender also has a file named calib1.py which has the calibration information of the potentiometer.



- Zigbee: The mechanical arm was attached to Spark V bot with Xbee chip installed. Thus Zigbee protocol was used for communication between mechanical arm and PC. Initially, we thought of a wired connection.

- Connecting Mechanical Arm to Spark V: The 5 potentimeters of mechanical arm are connected to 1,2,3,4,6 ADCs respectively. Microcontroller is written with a code to send the ADC pins via Zigbee on a signal interrupt.

5.3 Changes made in plan

- Initially we thought of implementing wireless communication between PC and the robotic arm. But the robotic arm did not have an Xbee component. So, we changed the design to communicate to the arm thorugh serial port, i.e., wired connection.

6 Future Work

- Wireless communication can be implemented in the robotic arm. Currently, we communicate with it through serial port. The system will be made more flexible.
- The whole robotic arm can be made mobile; to increase its functionality and area of contact. A separate control for its movement can be designed.
- The mechanical arm can be redesigned to fit on a human arm, so that it becomes even more easier and intuitive to control the robotic arm.
- The system can be redesigned as follows: first we record the movements of the mechanical arm in the simulation and then we allow necessary changes in it. Then, the whole recording can be replicated in the robotic arm. This will allow predefined and automatic movements in the Robotic arm.

7 Conclusion

We were able to realize the vision we started the project with. Our goal of intuitively controlling the robotic arm has been achieved in the project.

Some real world projects where it can be used:

- Automated Mason: The project can be used to build walls with much less effort.
- Games: Many games can be designed by using the mechanical arm as a robot, for example chess. Two player games involving competitions among two robotic arms can be designed.
- Bomb Diffusion: The robotic arm can be cotrolled from a safe place to diffuse a bomb.

8 References

- Screencast on installation of blender
- Blender Video Tutorials
- Projects on E-Yantra Website