Robot Herd

Swarm Robotics : Follow the Leader

ITSP 2017

INTRODUCTION:

The aim of this project was to build 3 - 4 robots which followed a single robot at the forefront (the "leader"). The robots would first be scattered about randomly in an open space and would then, through inter-communication, assemble themselves in a straight line, and then march along in a line.

DESIGN OF THE ROBOT:

We looked up many designs for swarm robots on the internet[1][2][3] and decided to settle on the Jasmine platform because of its simplicity and ergonomics. Since we decided to focus on only one particular application, we decided to trim the design in the Jasmine platform according to our needs, including removing remote controls, TWI interfaces.

It was decided that communications would be facilitated solely through IR devices. Each robot would be equipped with 6 such sensors. The only thing that remained was deciding on the protocols of communication. For this, we took inspiration from implementations of different applications, provided in the documentations of bot simulators[7][8][9].

The final components of each bot and their quantity as well as the price of each component:

S.No.	Component	Quantity per bot	Cost per item
1	LiPo 3.7 V 500mAh cells	2	150
2	Arduino Pro Mini 5V	2	265
3	L293D circuit	1	
4	Geared motors	2	133
5	Wheels	2	10
6	IR transmitters	6	
7	IR sensors	6	
8	Resistors (26K ohm)	6	
9	Resistors (470 ohm)	12	
10	Extras (wires, LEDs for modes, chassis)	-	

DETAILED REPORT:

- **Chose Arduino pro mini as our microcontroller due to its size and it's output voltage of 5v (which can also be used to power the H-bridge)**
- **Abandoned ultrasonic sensor in favour of only IR sensors for all purposes**
- **Initial idea for algorithm to assemble bots :**
    - **A bot starts emitting signals from all of its IR transmitters. Of all the bots that receive the signal, only one responds and relocates itself near the signalling bot. An ID for this bot is established. Now it starts emitting signals instead of the former and the process repeats. Needless to say, bots whose IDs have been established do not respond. After all bots have assembled, i.e., all bots have been assigned IDs, the bots can begin to move.**
    - **Problems: ensuring only one bot responds, establishing an ID which signifies the bot's position in the line**
- **Created a prototype of a bot with different material. The idea was to change to the original material (wood/metal) and get printed circuit boards once the prototypes have been confirmed to work as desired**
- **Update to algorithm for assembly:**
    - **To ensure that only one bot responds, a bot that receives the signal should send an acknowledgement signal in turn (which should be pulsed to differentiate it from any reflecting signals). Once the emitting bot receives the acknowledgement signal, it will emit signal only in the direction of the received signal.**
    - **Problems: arranging for the case where a bot obstructs the emitting signal, specifying the acknowledgement signal.**
- **Range of IR sensors were not suitable for our needs, so we decided to experiment with sound sensors for the moment after consulting with seniors. An alternative source, Ultrasonic sensors were also abandoned, since they require a precise line of sight**
- **Toyed with using sound sensors:**
    - **Each bot will be assigned a particular ID corresponding to a frequency which it will use, which will be known by all bots**
    - **When a particular bot emits a signal, the other bots can identify which ID is active and the bot which is required to respond. The direction can be pinpointed by source localisation[12]. After reaching near the bot, the second will start emitting its signal and first will stop.**

    - **Pros: can send a signal of different frequency for each bot so there's no interference.**
    - **Cons: generating a signal, receiving and identifying frequency, eliminating noise**
- **Turned out that the IR sensors we were using were faulty. Actually they give have a pretty good range, about 10 cm, so we decided to switch back to them.**
- **Added an L293D circuit as it presented a greater control over the motion as compared to directly hooking up the motor to the arduino via transistor at the cost of making a small circuit..**
- **Motors and sensors are up and running.**
    - **Working videos:**
        - **LED**
        - **Motor**
        - **Sensor**

Framework

- **The original framework we thought of using could not handle shear stress and the packaging area was very small. So we decided to try out wood as the lateral walls.**
- **A better alternative was found by increasing the size of the bot and keeping the source material the same. For handling shear stress, we decided to add a rigid L -shaped support at one of the edges.**

Motors

- **The motors were running correctly but started stalling when they were put on the ground. So we tried out different motors with different RPMs and torques and their combination with different wheel sizes.**
- **On calculation, we found that the torque required to move our bot is roughly 40 gcm. So we settled on small geared DC motors which provide more than enough torque.**
- **An unforeseen advantage was that due to its low power consumption, we do not need to provide an external power supply in parallel. (more on this later)**

Circuit design

- **During the initial testing of individual components separately, we had 2 PCBs, one for each set of transmitters and sensors. Placing 2 PCBs right on top of each other would present a problem in terms of both stability and wiring. So we redesigned the communication hardware and put all of it on a single PCB.**
    - **PCB design:**
- **Finally, we got a bot that was able to follow a simple IR signal anywhere.**
- **Added 2 leds as markers for distinguishing between different modes of the bot when powered by an external supply.**

Code

- **To take into account the different sensitivities of sensors, we converted the readings into a scale of 100 using the maximum and minimum values of the sensors.**
- **Update to algorithm:**
    - **We were unable to establish IDs with the algorithm. So now, we have switched to a different algorithm: We have 4 modes**
        - **0 - Idle, listening for communication**
        - **1 - Following a bot**
        - **2 - Transmitting signals for another bot to follow**
        - **3 - Standing by, waiting for all bots to assemble**
    - **This method does not call for any ID, so it is advantageous in a way that if any bot can be added to the current swarm without any additional changes in the code.**
    - **Mode 0 - Listens in all sensors, looks for communication on the sensor which is activated the highest. If it can communicate, it transitions into mode 1**
    - **Mode 1 - Follows along the sensor where it receives the lowest value below a certain limit. If it stays out of this limit for a very long time, it reverts to Mode 0. Otherwise, after following, if it reaches within a certain range, determined by the sensor value, it changes into Mode 2 and emits a signal to let the bot in front know it is there.**
    - **Mode 2 - The bot does not move and emits signal one-by-one along all transmitters. Whenever it accepts a communication, it sends signals only along that sensor, until the bot following comes within a range. It then changes into Mode 3. If after complete 3 cycles, it is unable to receive any communication, it is assumed to be the last bot and sends a signal back along the receiver.**
    - **Mode 3 - The bot waits to receive a signal from the bot behind it that the line assembling is complete. If it does receive such a signal, it transmits a signal to the bot in front, relaying the same message and changes into Mode 0.**
- **For the leader, the code is a bit different. It has no following mode and thus mode 0 and mode 1 are absent. Mode 3 now sends the bot into mode 2 instead of 0.**
- **Originally, for establishing a link between two bots, we had used a simple system: one bot sends a simple IR signal and other receives it alternately thrice. But this system proved to be a liability there were many false positives (also due to lack of time sync between the bots). So we had to use a more complex signal.**
- **For verifying that 2 bots are in-line, the bot in mode 2 kickstarts the verification process by sending a pulse along the line where the second bot supposedly is. The other bot recognises the pulse and sends back a signal. To let the second bot know**

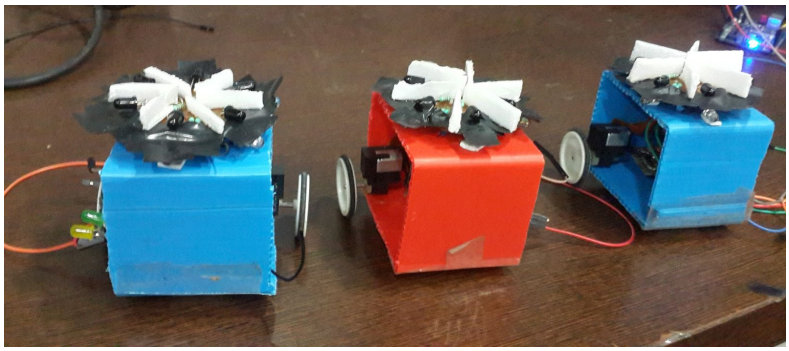
that it received the signal, a signal is sent back by the first bot. If at any step, the signals are not received, the process is terminated and a false value is returned.

- **Care had to be taken to ensure that when one bots emits the signal, the other one receives it.**

Circuitry (power consumption)

- **Originally, we used a 10 ohm resistor with the LED. But when we connected it to a battery, it drained off too quickly (12-15 seconds). The reason for using 10 ohms was to increase the signal strength to ensure greater coverage.**
  - **10 ohms drained the battery too quickly,**
  - **470 ohms proved to be a too high resistance. The signal strength reduced by too much.**
  - **250 ohms was the Goldilock's number. It doesn't consume too much power at once while providing a good signal.**
  - **Another factor which improved the battery life was that we had changed the algorithm so that only LED fired at a time.**
- **The motors which we finally used can be powered by the inputs of the IC alone without any external power supply. But since there were fluctuations the motor's performance, we decided to hook it up to the battery directly.**
- **The batteries which we required had to be small and by our calculation, with a capacity of 260mAh to run it for a reasonable time. Also, the Arduino required over 5 V of input. So keeping the cost low and the constraints, we decided on 2 small rectangular LiPo cells of 3.7 V each. The capacity of each cell we bought was 500 mAh, just to be on the safe side.**

- **Picture of the completed bots:**



VIDEOS

- **2 bots**
- **3 bots**

REFERENCES

---

1. **Jasmine swarm-robot platform : link**
2. **E-Puck : link**
3. **Kilobots : link**
4. **Boids : A comprehensive list of implementations of boids flocking (link)**
5. **The Convergence of Bird Flocking - Chazelle (link)**
6. **Synchronization of Pulse-Coupled Biological Oscillators - Morello, Strogatz (link)**
7. **ARGoS - Simulator of Swarmanoid project (link)**
    a. **Implementation of Light Synchronization (link)**
8. **BIOS, Communication in Jasmine (link)**
9. **Stage - robot simulator (link)**
10. **Arduino Microphone (link)**
11. **Generating Sine Waves-**
    a. http://web.csulb.edu/~hill/ee470/Lab%202d%20-%20Sine_Wave_Generator.pdf
    b. https://www.engineersgarage.com/embedded/arduino/variable-frequency-sine-wave-generation-using-arduino
12. **Acoustic Source Localisation (link)**