Project 1 Final Report

Due Monday, February 16, 2004

Group 8:

Nouman Khalid Jan

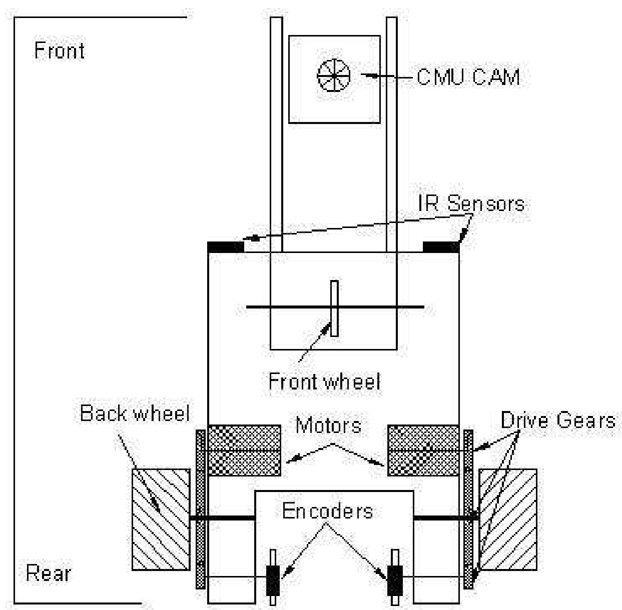
Andy Lau

David Goldberg

Robot Design

Name of Robot: “Badmaash” (The Out law)

Robot Schematic Diagram



Design Phase:

The initial design of the robot was inspired from the *Handybug9719* described in Fred G. Martin’s Robotic Exploration, chapter 2. The robot was performing well during the preliminary trials, but due to the structural flaws in the implementation, the *Handybug9719* design was not a good idea.

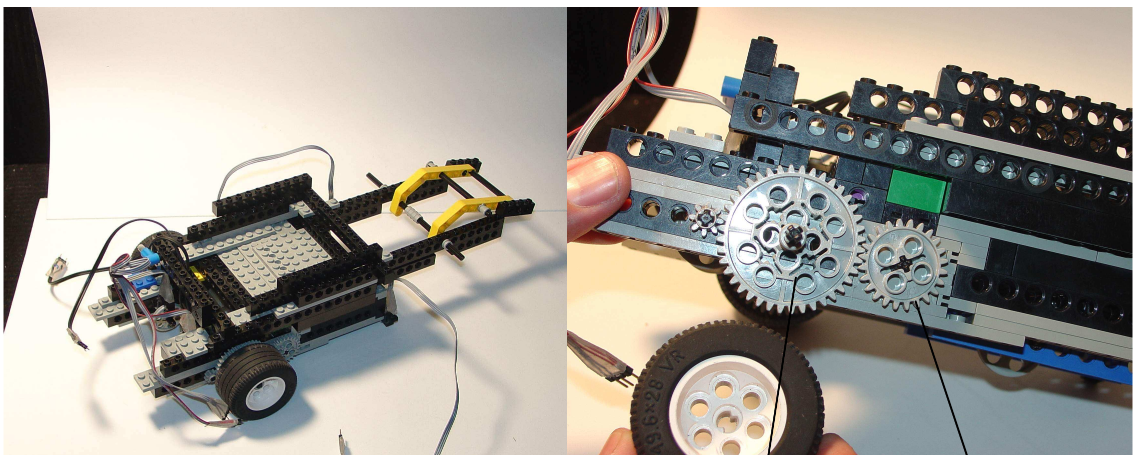
The final design of the robot has a cantilever arm in the front to hold the CMU cam, which detects colors. The front wheel on the robot is somewhat in the center, to make the robot to turn on the same axis. The two motors are used to drive individual wheels. The encoders are installed at the back to help in the right, left, and about turns.

Drive Train and Gears:

The main power for the robot was supplied by two motors, which drove each wheel individually. The gear ratio maintained was 1:1.67, which made it possible for the motors to transfer maximum horsepower to the wheels.

The wheels used initially were thin, which caused low traction between the surface of the tiles and wheels. Using a thicker wheel solved that problem. The base of the robot was quite heavy, and due to the max horsepower transfer, the only problem that was encountered was the robot not stopping at the required point. So to tackle this problem, the motors were run in backward direction to counteract the momentum of the robot.

In the picture the driving gears and the robot is shown with out the Handy board installed.



The Gears are shown in the figure.

Motor Gear

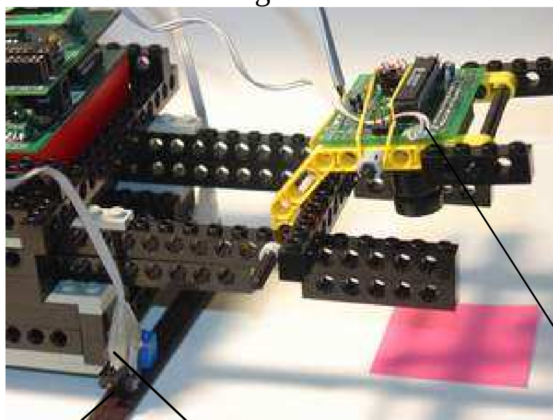
Wheel Gear

The front wheel of the robot was located approximately in the center of the robot so the robot would turn on its axis. The front wheel could not move sideways, which caused the sideways drag. The drag encountered while making turns was overcome by the high revolution of the rear wheels.

IR Sensors:

The IR sensors were installed in the front of the robot, as you can see in the Design Schema. The IR sensors were used to track the black tape around the one foot block in the actual demonstration. IR sensors were placed in such a way that when the robot stopped after sensing the tape, the camera should have a good view to track the color, i.e. the CMU cam should be over the color.

The IR sensors were also used to align the robot on the black tape.



IR Sensor

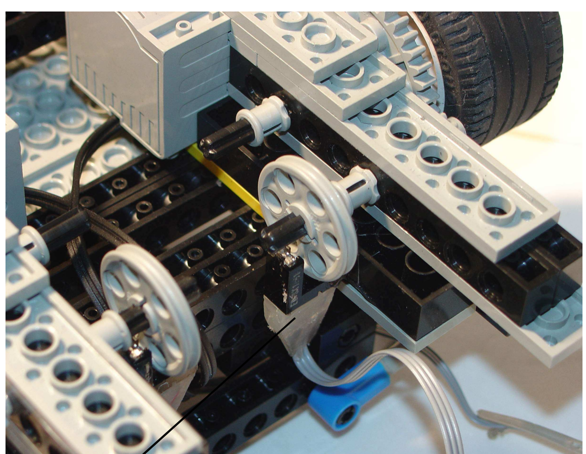
CMU Cam

Black Tape

Encoders:

The encoders were not in the initial design consideration, but as the implementation went on, it was necessary to make accurate turns, the encoders were used to solve the problem. The gear for the wheel on which encoders were installed was driven by the rear wheel gear with the ratio of 1:5.This produced appropriate revolutions and ticks that the encoder counted in order to make sure the turn was complete.

Encoders were also used to align the robot when the IR sensors sense the black tape.

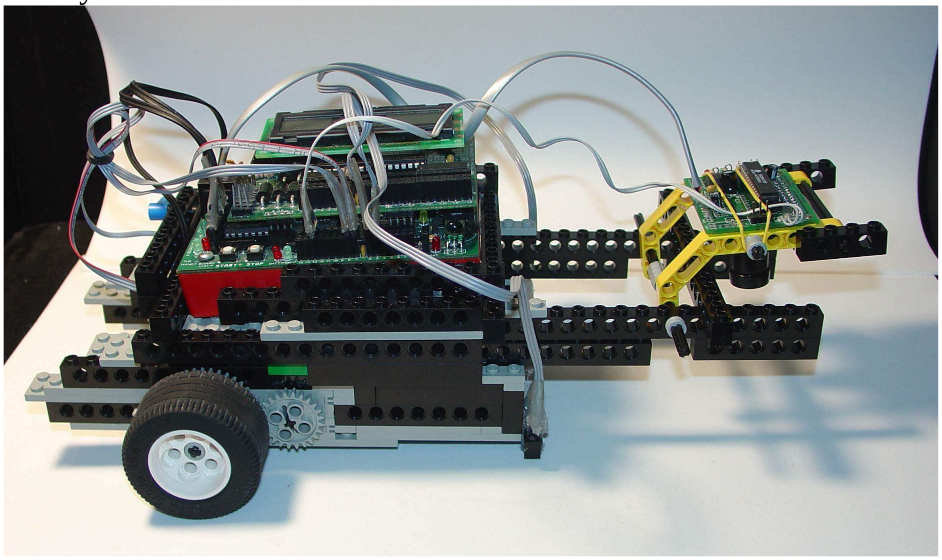


Encoder.

CMU Cam:

The CMU cam was installed in the front on a cantilever Lego beams. The CMU cam was placed in such a way that when IR sensors stopped the robot, the CMU cam was able to track the color.

In the initial design, the CMU cam was in a tilted position. In the end, it was decided that the CMU should face completely downward to allow the robot track the colors perfectly.



Design Flaws:

1. Due to high Horsepower transmission from the motors to the wheels the energy efficiency of the robot was great. The battery ran for a long time
2. The CMU cam was adjusted and installed in such a way that it was very easy for the camera to detect colors.
3. The robot was running fast with more momentum, which made hard to stop quickly.
4. The CMU cam had problem with shading because when it was over the color card, it blocked some of the light.
5. The front wheel could not move sideways which caused side ways drag during turning.

Special Thanks:

Dr Dean Hougen.
Pedro Diaz.
Group 5.

Robot Code

```
#use "cmucamlib.ic"

/*.....................................................................................
  Void Straight()
  turn on the motor to go straight and calibrate itself while doing that
.....................................................................................*/
void straight ()
{

    motor(4,19);
    motor(1,16);
    while ((analog(5) < 120) || (analog(6) < 120))
      {}
    //   motor(4,-10);
    //   motor(1,-10);
    //   sleep(0.3);
    ao();
    b();

    sensingcolor();

    //     ao();
}

/*.......................................................................................
    void turn left()
    make the robot turn left
.......................................................................................*/
void turn_left()
{
    int L=121;
    enable_encoder(0);
    motor(4,40);
    motor(1,-40);
    while(read_encoder(0) <= L )
      {
    }
    ao();
    sleep(1.0);
}
/*........................................................................................
   void turn right()
........................................................................................*/
```

```
void turn_right()
{
   int L=117;
   enable_encoder(1);
   motor(4,-40);
   motor(1,40);
   while(read_encoder(1) <= L )
     {
   }
   //sleep(2.3);
   //    reset_encoder(1);
   ao();
   sleep(1.0);
}

void go()
{
   int L=45;
   enable_encoder(1);
   motor(4,19);
   motor(1,16);
   while(read_encoder(1) <= L )
     {
   }
   //    reset_encoder(1);
   //sleep(2.3);
   ao();
}
/*..........................................................................................
  void about_turn()
..........................................................................................*/
void about_turn()
{
   int L=228;

   motor(4,-11);
   motor(1,-11);
   sleep(1.7);
   ao();
   sleep(.2);

   enable_encoder(0);
   motor(4,40);
   motor(1,-40);
   while(read_encoder(0) <= L )
     {    }
```

```
    //sleep(2.3);
    reset_encoder(0);
    ao();
}

void b()
{
    motor(4,-2);
    motor(1,-2);
    sleep(0.7);
    ao();
 }
/*..........................................................................................
..........................................................................................*/
void sensingcolor()
{
    sleep(.1);
    if(trackRaw(30,65,125,185,75,130)>0)
      {
        printf("blue \n", track_confidence);
        go();
        turn_right();
        straight();
    }
    else if (trackRaw(125,185,45,65,13,19)>50)
        {
          printf("orange \n", track_confidence);
          go();

          turn_right();
          straight();
      }

      else if (trackRaw(100,190,12,40,12,20)>0)
          {

            printf("pink \n", track_confidence);
            about_turn();
            straight();
        }
        else  if (trackRaw(140,180,140,180,13,19)>0)
            {
              //   alloff();
              printf("yellow \n", track_confidence);
              beep();beep();beep();beep();
              kill_process(sensingtape());
```

```
            sleep(2.0);

        }
        else  if(trackRaw(95,120,135,170,15,25)>0)
            {

              straight();
              printf("green \n", track_confidence);
          }
          else
            straight();
}
/*...........................................................................................
..........................................................................................*/
void sensingtape()
{

    if( (analog(5)>170 )|| (analog(6)>170) )
      {

        b();
        sensingcolor();          //should be put after right_cali and left_cali
        if ((analog(5)>170) && (analog(6)<170))
          {
            ao();
            right_cali();
        }
        else if ((analog(6)>170) && (analog(5)<170))
            {
              ao();
              left_cali();
          }
          else
            {

              sensingcolor();
          }
    }

    //sleep(.01);
}
/*...........................................................................................
void right_cali()
adjust right side to where the tape is
...........................................................................................*/
```

```
void right_cali()
{
   motor(1,-5);
   motor(4,10);
   while (analog(6)<170)
     {}
   //  motor(1,47);

   b();
   //alloff();
   sleep(0.5);
   sensingcolor();
}
/*...............................................................................
void Left_cali()
adjust left side to where the tape is
..............................................................................*/
void left_cali()
{
   motor(4,-5);
   motor(1,10);
   while(analog(5)<170)
     {}

   sleep(0.5);
   sensingcolor();
}

//  rgb Camera *****This method was given to the team by Group 5!

int clamp_camera_rgb()
{
   int i;
   printf("Point at white  and press start\n");
   while (!start_button());
   send_R_command_require_reply("CR 18 44\r",9);
   //    for (i= 15; i > 0; i--) {
   //      printf("Setting white   balance %d..\n", i);
   //      sleep(1.0);
   //    }
   send_R_command_require_reply("CR 18 40\r",9);
   printf("Done\n");

}
```

```
/*.............................................................................................
void
*/
void main()
{
    while(!start_button())

      init_camera();
    clamp_camera_rgb();
    //  clamp_camera_yuv();
    straight();


    start_process(sensingtape());

}//main
```

Robot Code Documentation

Use of sensors in Code

IR sensors : The IR sensors were used to sense the black tape and align the robot in the original design so IR sensors were put in front to detects the tape and make adjustment there. But it was not possible for the robot to be aligned while moving and not sensing the black tape so they were not being used. In the later design it was much more convenience to use encoders instead to align the robot while moving straight which was much more accurate.

Encoder: The encoder ticks were used to make turns. The program was counting certain amount of ticks from the encoder confirming a left, right or about turn. This idea was the best as we can easily adjust the number of ticks to see the variation in turning.

CMU camera-
We used the built-in function track_Raw to get the confidence level while reading a color.

On the final testing, a day before demonstration the camera had a problem in recognizing the orange and pink color and making every possible effort we could not fix the problem. I would like to thank Group 5 for their special help that helped us switching from yuv to rgb mode. The values the track colors were provided by them.

Algorithms

Straight()
This function called the two motors in the robot to go forward with speed 19 on left and 16 on the right side to make it go straight until it sees the black tape. Whenever they sensed the black tape it will turn the motor off and call the function b to brake more dramatically. After that it calls the sensingcolor function.

Turn_left()
It will set the number of clicks that needed to make a proper left turn. Enable the encoder on left and call the left motor to backward with speed 40 and right motor 40. this will turn until the number of clicks have been reached. Motor will be off and turn sleep for 1 second.

Turn_right()
It will set the number of clicks that needed to make a proper right turn. Enable the encoder on right and call the right motor to backward with speed 40 and left motor 40. this will turn until the number of clicks have been reached. Motor will be off and turn sleep for 1 second.

Go()
This is another function that will make the robot go straight until the encoder on right sense the black tape.

About_turn()
It makes the motor turn backwards for 1.7 seconds, then turn off motors and sleep for .2 seconds. Turn on the left encoder. These timings are used to make proper adjustment. The motor on right will set to speed 40 and motor on left will be -40 to make a 180 degree turn. It will continue to turn until certain number of clicks has reached

B()
This function is basically used to stop the motors. It will set both of the motor to speed -2 and tell it to sleep for 0.7 seconds and all motor off. This function is needed because the alloff function doesn't brake fast enough because of high momentum whenever IR sensed the tape.

Sensingcolor()
At first it will go to sleep for .1 second so it will have a little bit more time to read the color of paper. A number of if statements and the track Raw functions are called to used to tell the robot what to do when it sees these color. If the camera sees the blue color it will print the word blue in the LCD screen on handy board and it will call the go function, turn_right function and at last the straight function. Do almost the same thing according to what color it sees. If none of the color are found it will call the straight function again

Sensingtape()
If either of the IR sensors senses the black tape it will call the sensing color function. And if the left one senses the black tape but the right one doesn't senses it, it will call the right_cali function to align the robot. And if the IR sensor on the right sense it first but the other doesn't it will call the left_cali function to align so both of the IR sensors are on the black tape.

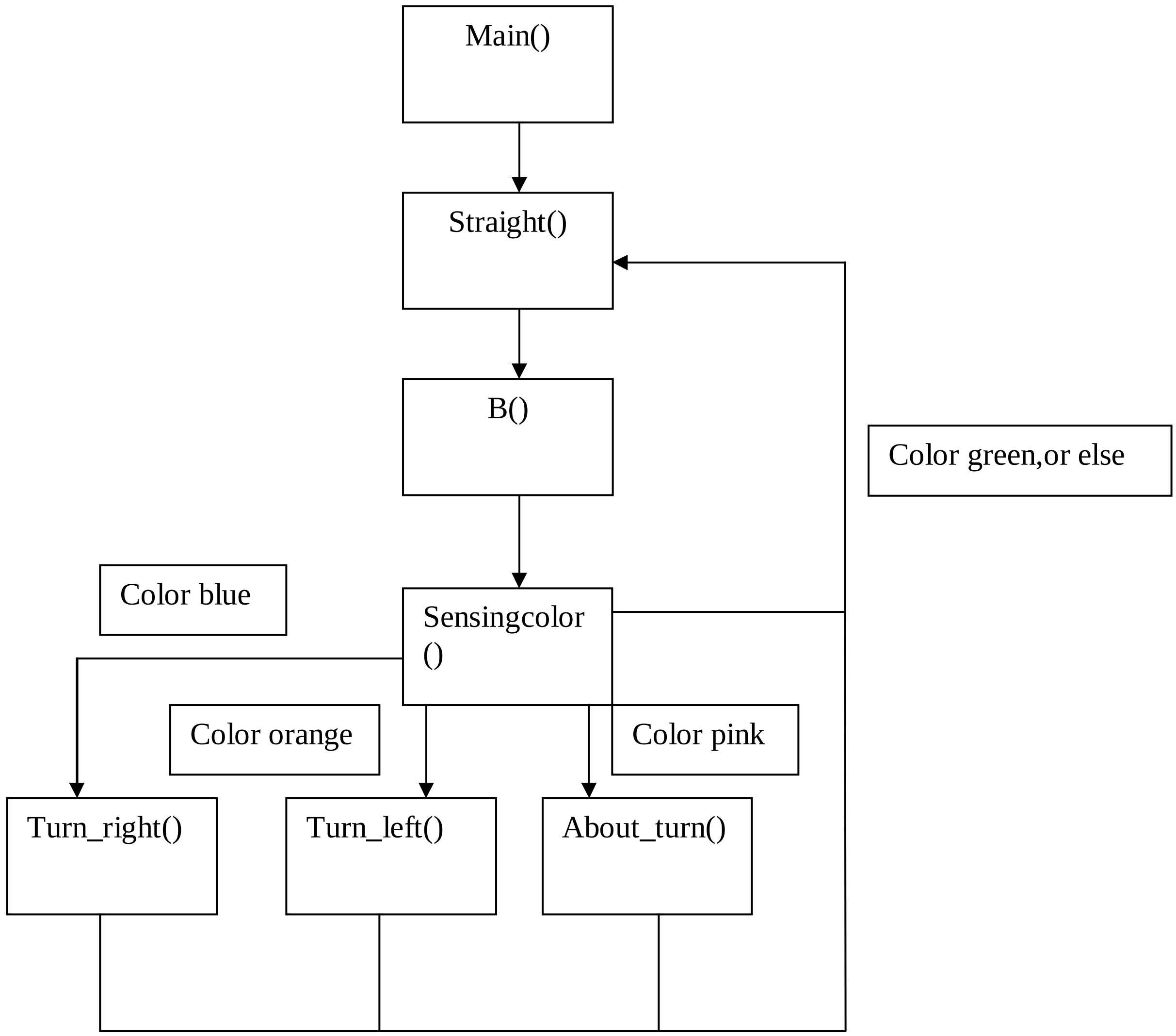
Right_cali()
It will stop the left motor until the IR sensors on right reached the black tape, and then it will call the sensingcolor function.

Leftcali()
It will stop the right motor until the IR sensors on left reached the black tape, and then it will call the sensingcolor function

Clamp_camera_rgb()
I would like to thank Group 5 which helped us in writing this function .The built in function clamp_camera_yuv() wasn't able to sense pink before. It worked with this function.

Main()
It called the go straight function and starts the process on sensingtape.

Flow chart below.



Data Structures

This was a reactive design and there were no data structures used in the code.

Team Organization Evaluation and Plans

Although this project was technically challenging, it presented equal challenge in coordination and planning for the team members. One constraint was that the robot could only be taken home by one person at a time. This was coupled with the fact that one would have had to have been extremely creative to get any work accomplished without the robot present to experiment with. Another constraint was that, with two group members living over 45 minutes away, the environment in which the robot was to perform was not readily accessible (We also did not have flat tiles in our homes). Thus, adjusting the camera, going straight, and turning was not possible to execute in our homes. Then, when we all met during our time that we had free on Monday and Wednesday on campus, we only had loose guidelines for tasks to accomplish. This meant the time we spent together was not neccesarily spent on the right goals. Further, usually only two out of the three of us was working at any given time, and even those two could not think clearly being that two other members were present. Also, since two were working simultaneously and each person has only so much patience, the other could not isolate problems and think clearly for a long period of time without the other wanting to change the direction of solving the problem before the previous direction could bear fruit. The members were amicable towards one another throughout the entire process, however. These frustrations, fortunately, were not allowed to become personal conflicts, but remained snags along the way to the goal.

After completing this project, the group members learned about each others work habits that they will use for organization plan for project two. For example, David prefers to make significant progress early on, building his confidence and creating momentum. He became discouraged when one week before the ultimate deadline, the project seemed no where near completion and little progress seemed to be made. Nouman, however, is motivated by deadlines and feeds off the energy they create. Though he would prefer to get things done ahead of time as well, he is comfortable with working long hours right before the project is due, so he worked all night long before the deadline. Although in one sense this conflicting style of achieving goals could create difficulties, it can also be used to the groups advantage. In the future, David can be sure and make significant progress in the beginning and middle of the work time, while Nouman can get significant work done towards the end of the project.

In creating the organization plan for this project, the members did not ask each other about their inclinations to work harder long before the project is due versus working harder towards the deadline. This is an important issue that should have been clarified and will be in the future, in order to use it to the groups advantage.

One group member, David Goldberg, found the he needs to alter his individual time management scheme completely. At the beginning of this project, he worked on each individual class until the necessary work in it was completed, only then moving on to the next class. Because of the magnitude of work involved in this project, he became discouraged when one weekends worth of work did not have him almost finished with the project, since this already meant getting behind in other classes. Thus, his new plan is to work on one hour rotations of each class. In this way, he will not always be "putting out the flames" of work that needs to be done in every class. He will then feel more at ease with the time he is putting in on the project. An added advantage is that after waiting three hours to return to the project, he will have a new perspective on the issue at hand, helping him invest his time more wisely (on the more critical tasks).

If the team could do the project over again, the members would be more insistent on making progress from the day they received the lego kit and the assignment. The members underestimated the complexity of the assignment. They did not realize that so many aspects of the project needed to be tested and decided, rather than just implemented. Of course, this takes much more time. Futhermore, the members would create more goals that reflected the robot accomplishing tasks necessary to reach the final goal. For example, a goal should have been created simply to make the robot go straight. At the same time, the members would continue with the democratic organization of the group. This process worked even better than expected, since members naturally developed specialties (i.e. organization and teamwork analysis) that recurred throughout the project. Each person eagerly volunteered to continue contributing in that area since he already had experience.

Though there are multiple areas of improvement needed in the group organization, it should be noted that the group succeeded in its goal of making the robot run the course. Thus, the members feel positively about this project and are ready to tackle the next task with improved organizational and group skills!