S.C.A.R.A.B.

Search Capture and Rescue Air Boat

Marc Stracuzza
4EG-CEN
IMDL
Spring 1998
Professor: Dr. Arroyo

Table of Contents

Abstract

This report describes the IMDL project design SCARAB, an autonomous water platform robot. The report will cover the platform design, system functions, platform actuation and sensors, and the behaviors of the platform. The design process will be described along with the successes and failures encountered along the way to completing an autonomous water platform robot.

Introduction

Designing an autonomous water platform holds several distinct problems from land platforms. A water platform must be completely sealed from the entrance of water into the circuits. A water platform must also be stable in that a wave or gust of wind must not be able to flip the platform onto its side where water can damage circuits. The platform must be buoyant and thus must be light with large flotation devices. The platform must be able to sense and interact with objects under direct sunlight and all of the forces of nature since most of the uses of water platforms would apply outside. These problems are the basis upon which the robot platform was designed. Solving these problems are what made this platform both difficult and unique. The SCARAB design platform is meant to perform search and capture functions in an outdoor environment on open water. The search and capture function involves sensing a distress beacon, pinpointing its location, traveling to that beacon, capturing the beacon, and returning it to a "home base" area avoiding any obstacles that may hinder this function along the way. The platform uses IR emitters and detectors to do basic obstacle avoidance and uses sonar for beacon detection and location.

Mobile Platform

The SCARAB platform design is the main area that separates a water platform from a land platform. The body design cannot be flawed or unstable in any way. It must be buoyant, light weight, waterproof, and stable enough to resist the forces that nature can apply.

The main body is 11" in length, 11" in width, and 4" high. The front of the body has two angled sections to provide some aerodynamics. The top to the body is a completely separate piece that can be locked in place by a series of latches. This is to provide easy access into the circuitry that will be placed inside the main body.

The platform design chosen for SCARAB is a three legged structure. The main body is supported by two side legs offset towards the back of the body and angled outwards for more stability. A front leg is centered and offset to the front of the body and angled slightly forward. All three legs attach to the flotation device.

The flotation devices consist of two plastic 24oz 7-Up soda bottles. The bottles provide a sturdy and buoyant flotation device that are easily found and replaced if necessary. The bottles are placed side by side and connected to each other by a top piece which in turn connects to the legs.

The material used for the main body was Plexiglas which turned out to be a very challenging material to work with. There were no proven methods in cutting, assembling, or drilling the material. As it turns out, once the right method was found the material was very easy to work with and provided a sturdy, waterproof base to build a platform.

SCARAB also has a SONAR beacon that is a separate part of the project. The beacon is also made of Plexiglas with two 20-oz Coke bottles as its floatation device. It has three tears with the first tear holding the battery, the second holding the SONAR amplification circuit, and the top tear holding the TJ processor that drives the 40kHz signal. The beacon is approximately 12" tall and 8" wide.

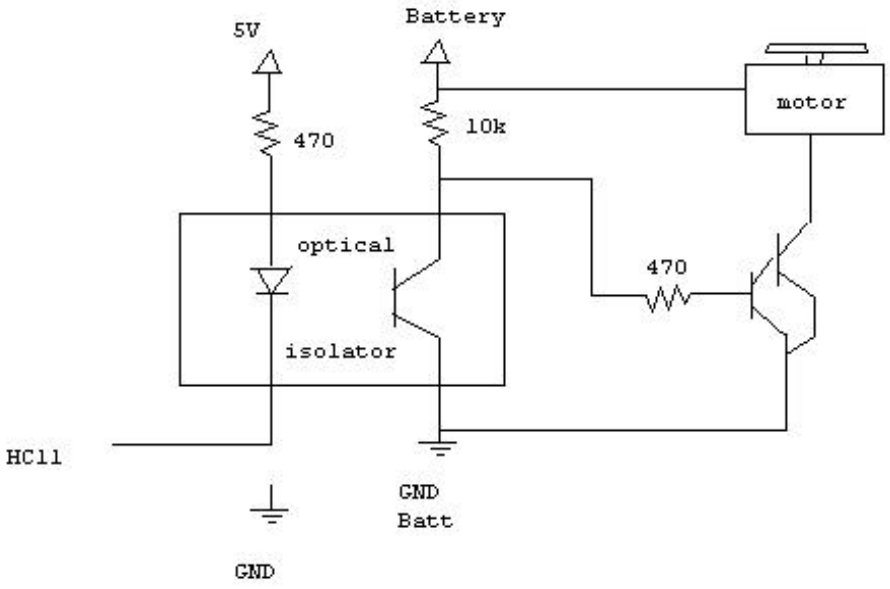
Actuation

The method used to drive the platform was almost too easily defined and applied. The idea to use two fans to provide the force needed to propel the platform was seen from the start. The motors necessary to do this needed to be gear-less. The lab was in possession of the exact type of motor needed and provided them towards the design.

The motors were constructed into fans by using propeller blades of model airplanes. The propellers were mounted using collars that attached themselves to a short shaft extended from the motor. The sizing of the collars took some effort but the design worked and the fans provided a good deal of thrust.

In a brainstorming session Aamir Qaiyumi conceptualized a way for two fans to provide easy turning and forward motion. The method was to place the two fans orthogonal to each other. Thus when both fans were on the vectors would cancel and cause a straight vector directly in between the fans. When one fan was turned off the vector would be to the side and cause the platform to rotate around the intersection of the orthogonal basis of the fans. The design worked perfectly the first time.

The fan motors did have one downside to them, they draw a lot of current and drain batteries at a fast pace. To drive these motors a special circuit had to be constructed and with the help of Scott Jantz the following design was incorporated and worked:

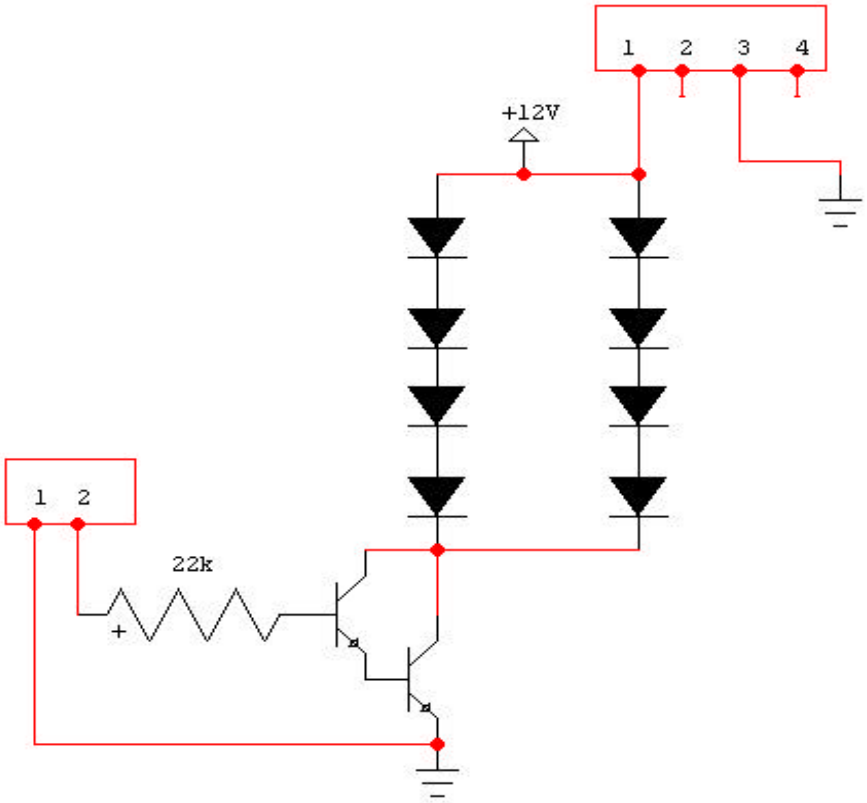


Sensors

The SCARAB platform uses IR, sonar, and a microphone. The IR is used for obstacle avoidance. The sonar is used to sense and locate a beacon. The microphone is used as a collision sensor.

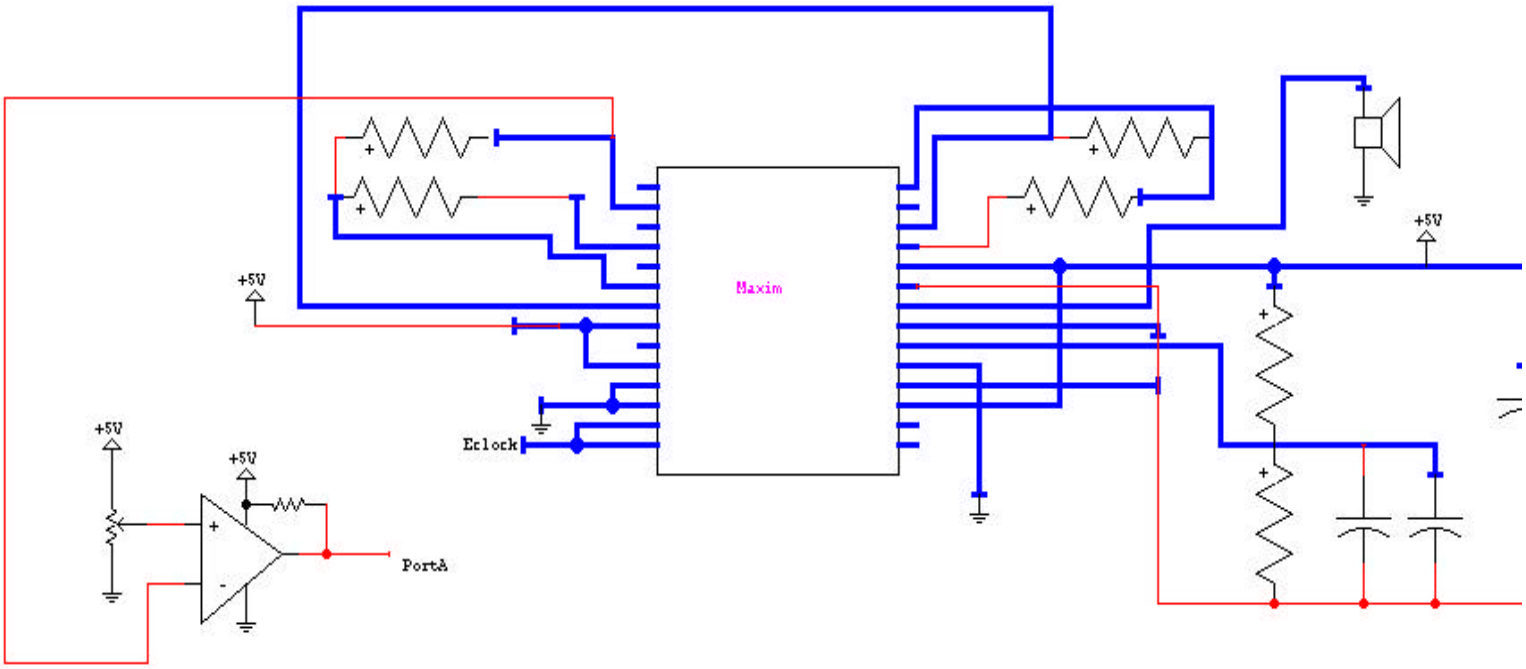
The IR is an amplified circuit that provides 330mA of current through ten IR LEDs. Two IR detectors are placed at the front end of the platform to either side of the "IR cannon" which extends slightly off the center of the front of the platform. The amplified IR circuit is necessary to avoid the interference caused by normal outdoor IR.

IR circuit:

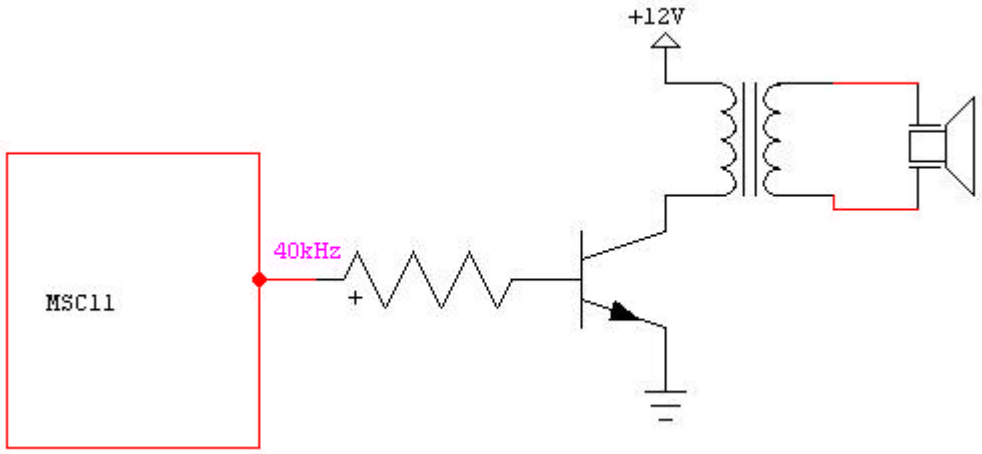


SONAR is also used on SCARAB. Two receiver circuits are placed on the front angled section of the main body. The two receivers allow for a pinpointing of the transmitted signal. The frequency of the signal is 40kHz. The transmitter board is mounted on the separately constructed beacon platform. The receiver and transmitter boards were designed by Frank Bond.

Receiver:

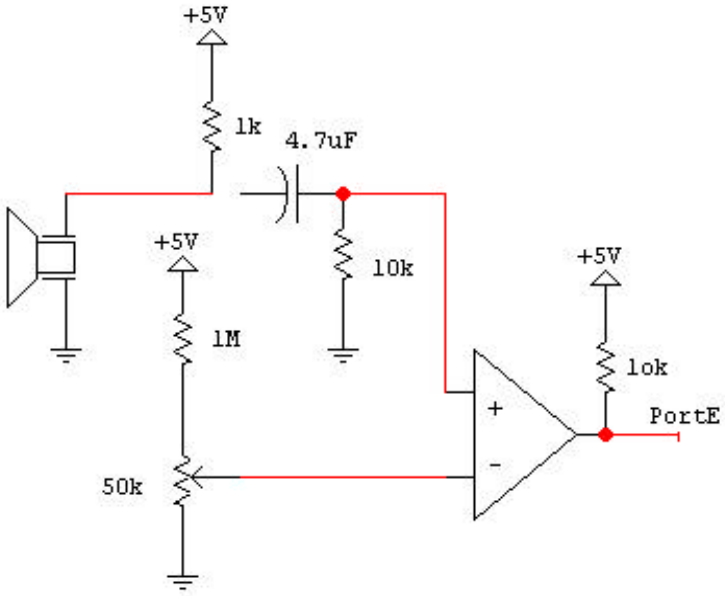


Transmitter:



A microphone is used as a simple collision detector. The microphone is placed inside one of the 7-Up bottles at the front of the platform. When a sound spike is sensed it indicates the detection of a collision.

Bump Sensor:



Behaviors

The behavior of SCARAB is rather straight forward. After launching from the shore SCARAB wanders around avoiding any obstacles until one of two circumstances occur. Either a distress signal or a collision is detected.

If a distress signal is detected the robot will locate the signal and head towards it until it reaches the source of the signal. The robot will still avoid obstacles while searching out the beacon. Once the beacon had been located the robot “captures” the beacon and then continues to avoid obstacles until retrieved.

If a collision is detected SCARAB shuts its motors down, the waits for approximately 2 seconds. It then puts full power to one motor to turn for approximately 2 seconds. Then it continues with obstacle avoidance and beacon locating.

Conclusion

SCARAB did not do all of the functions originally intended for it, but it did do most of them. The SONAR, IR, and bump sensor all worked. The project was a success and a lot of fun. I learned more from building this robot then anywhere else. I could not imagine a better circumstance or learning environment. I only hope that many other students take advantage of this class and gain as much from it as I did.

Coding

```
int IRright, IRleft, flipbit;
float IRmotorleft, IRmotorright;
int IRcounter;
int mic, bump, bump_count;
int sonarleft, sonarright, sonarhit, sonarleftmin, sonarrightmin, s_count;
float sonarmotorleft, sonarmotorright;

void main()
{
start_process(irhandling());
start_process(sonar40handling());
start_process(bumphandling());
start_process(arbitrator());
}

void irhandling()
{
flipbit=0;
poke(0x7000, 0x00);

while (1)
{
if (flipbit==0)
 {
   poke(0x7000, 0xff);
   IRright=analog(0);
   IRleft=analog(1);
   poke(0x7000, 0x00);
   flipbit++;
 }
else
 {
  flipbit=0;
 }

 if (IRright>112 && IRleft>112)
        {
         IRmotorleft=0.0;
         IRmotorright=0.0;
        }
 else if (IRright>100 && IRleft<112)
        {
         IRmotorleft=100.0;
```

```
        IRmotorright=0.0;
       }
 else if (IRleft>95 && IRright<115)
       {
        IRmotorleft=0.0;
        IRmotorright=100.0;
       }
 else
       {
        IRmotorleft=68.0;
        IRmotorright=75.0;
       }
 }
}


void sonar40handling()
{
sonarhit=0;
sonarleftmin=255;
sonarrightmin=255;
while(1)
 {
 sonarleft=analog(3);
 sonarright=analog(4);

 if (sonarleftmin<170 && sonarrightmin<170);
     /* sonarhit=1;*/

 if (sonarleft<sonarleftmin)
       sonarleftmin=sonarleft;
 if (sonarright<sonarrightmin)
       sonarrightmin=sonarright;

 if (sonarleft==255 && sonarright==255)
       {
        sonarmotorleft=IRmotorleft;
        sonarmotorright=IRmotorright;
       }
 else if (sonarleft>sonarright)
       {
        sonarmotorleft=70.0;
        sonarmotorright=0.0;
       }
 else if (sonarright>sonarleft)
```

```
        {
         sonarmotorleft=0.0;
         sonarmotorright=70.0;
        }
 else
        {
         sonarmotorleft=50.0;
         sonarmotorright=50.0;
        }
 }
}

void bumphandling()
{
while(1)
 {
mic=analog(2);

if (mic>128)
        bump=1;
else
        bump=0;
 }
}

void arbitrator()
{
while(1)
{
if (bump==0)
        {
        if (sonarhit==1)
         {
         s_count=0;
         while (s_count<200)
                {
                motor(0, 0.0);
                motor(1, 0.0);
                 s_count++;
                }
         while (s_count<400)
                {
                motor(0, 0.0);
                motor(1, 100.0);
                s_count++;
```

```
            }
        s_count=0;
        sonarhit=0;
        }
       else if ((IRmotorleft>69.0 && IRmotorright>69.0) || (sonarleft<250 ||
sonarright<250))
        {
         motor(0, sonarmotorleft);
         motor(1, sonarmotorright);
        }
       else
        {
         motor(0, IRmotorleft);
         motor(1, IRmotorright);
        }
       }
else
       {
       bump_count=0;
       bump=0;
       motor(0, 0.0);
       motor(1, 0.0);
       while(bump_count<200)
            {
            motor(0, 0.0);
            motor(1, 0.0);
            bump_count++;
            }
       while(bump_count<400)
            {
            motor(0, 100.0);
            motor(1, 0.0);
            bump_count++;
            }
       bump_count=0;
       bump=0;
       }
}
}
```