

RoboRugby 2010

Robotics Design Project
EEEN 10020

Lecture 3 - Lines, Beacons & Timing



UCD School of Electrical,
Electronic and Mechanical
Engineering

Scoil na hInnealtóireacta
Leictirí, Leictreonaic agus
Meicniúla UCD

Progress so Far ?

- **Competition strategy design under way**
 - your team should have a few plans under discussion
 - consider strengths, weaknesses, threats
 - report on design process needed next week
- **Robot able to detect collisions**
 - react differently to collisions on left and right
 - get away from obstacles
 - count number of obstacles hit
- **Robot able to detect white lines on table**
 - react to line in simple ways



2

This Week...

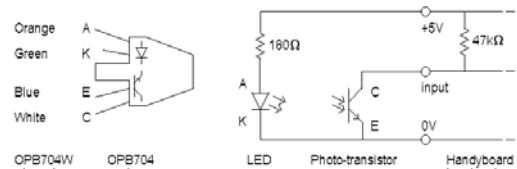
- **Review Challenge 2 software (in tutorial)**
 - need to be able to do one thing repeatedly
 - when some condition met, move on to next task
- **Learn to follow lines on table**
 - useful for competition - many balls on lines at start
 - develop algorithm for following lines
- **Learn to use beacons**
 - infra-red signals from ends of table
 - guide robot to scoring area
 - develop algorithm for driving towards beacon
- **Combine the two - Challenge 3**
 - follow line until hit something
 - then find and drive to beacon



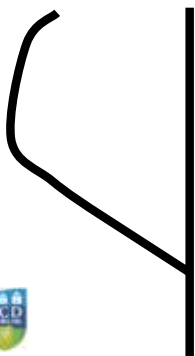
3

Line Sensor

- **How does it work?**
 - infra-red LED emits light at angle
 - photo-transistor detects reflected light
 - must be close to table, so narrow field of view
- **How to use it?**
 - output is voltage: higher = darker
 - connect to analogue input port
 - analog() function returns integer: higher = darker



Line Following



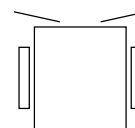
- **Robot should drive along line**
 - line may have corner or curve
 - consider effect of junctions
- **Sensors**
 - how many needed?
 - where to place on robot?
- **Algorithm?**
 - this week - simple algorithms, using one or two sensors...
 - assume already on line at start
- **How to get started?**
 - drive to line and turn to follow...



5

Line Following - One Sensor

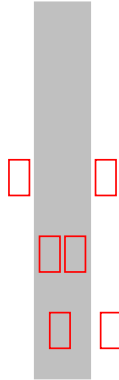
- **Algorithm?**
- **Where to put sensor on robot?**



6

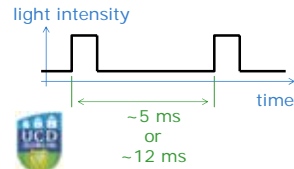
Line Following - Two Sensors

- Relative position of sensors:
 - how far apart?
 - what state is "on line"?
 - interaction between sensors?
- Algorithm?
- Junctions?
- Three sensors ?



7

Beacons



8

- Beacon transmitter
 - one at each end of table
 - transmits pulses of infra-red light
 - different pulse intervals: approx. 5 ms, 12 ms
- Beacon receiver on robot
 - detects signal
 - measures repetition time
- What use is it?
 - can guide robot towards correct scoring area
 - can allow robot to orient itself on table
 - used to start matches ! !

Beacon Receiver



- Internal and external receivers available
 - can put external receiver high on robot
 - can use either or both at different times
- Need shield - restrict horizontal angle of view
 - also keep out room or table lighting from above
- We provide assembly-language software
 - runs in background, measures pulse interval
 - functions to control receivers, get time interval



9

Beacon Receiver Software

```
#use beacons.icb // use the beacon software
// put at start of program, before any functions
```

```
beacon_init(1); // turn on internal receiver
beacon_init(2); // turn on external receiver
beacon_init(3); // turn on both receivers
beacon_init(0); // turn off both receivers
// load on processor - only enable receiver needed!
```

```
btime = beacon(1); // get interval from internal
btime = beacon(2); // get interval from external
// returns integer: time interval in microseconds
// returns 0 if no signal or receiver turned off
```



10

checkbeacon() function

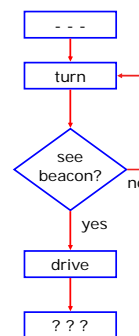
```
/* Function to check beacon time, make decision.
Argument 1 => internal receiver, 2 => external.
Returns 0 if no signal, 5 or 12 if beacon seen.
Assumes beacon receiver already enabled! */
```

```
int checkbeacon(int beaconReceiver)
{
    int btime = beacon(beaconReceiver);
    // create variable and get beacon time
    if ((btime > 11000) && (btime < 13000)) return 12;
    // is it approximately 12 ms?
    else if ((btime > 4000) && (btime < 6000)) return 5;
    // is it approximately 5 ms?
    else return 0; // if neither, return 0
} // end of checkbeacon function
```

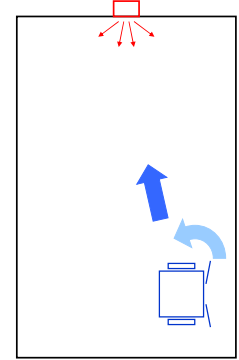


11

Find Beacon and Drive to it



12



Simple Approach - Dangerous!

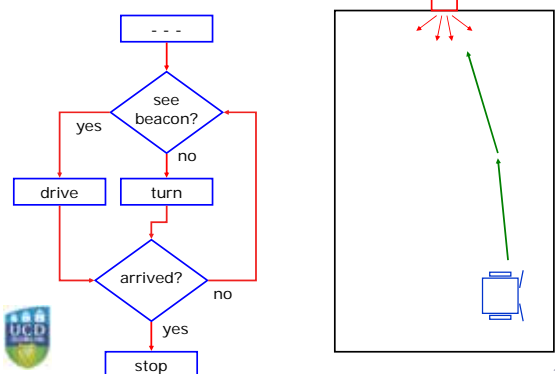
```
int mybeacon = 5; // define which beacon
...
motor(1, 50); // turn, looking for beacon
motor(2, -50);
while (checkbeacon(2) != mybeacon) { }
// empty loop - wait until see beacon
motor(2, 50); // drive straight (towards beacon?)
```

- Danger: no checking as drives towards beacon...
 - might not be perfectly aligned when see beacon
 - might not drive in straight line
 - might hit something
- Danger: if beacon not seen, spins forever



13

Better Solution - still flawed



14

Time Limits

- What if beacon signal blocked?
 - simple solutions turn forever...
- Should stop after reasonable time
 - then drive to new place and look again?
- Many similar problems in robot software
 - finding line
 - following line
 - hunting for balls
- Need to put time limits on loops
 - decide in advance how long to spend
 - if no success, move on to plan B
- Every while() loop should check the time!
 - need to cope with unexpected events...



15

Timing Functions

- float seconds(void)
 - returns system time in seconds, to nearest ms
- long mseconds(void)
 - returns system time in milli-seconds
- void reset_system_time(void)
 - sets system time to zero
 - will **not** be available in competition
- Typical example:
 - define variable to hold time limit
 - float limit;
 - before start of loop, set limit at point in future
 - limit = seconds() + 7.5;
 - compare time with limit in loop test
 - while (??? && (seconds() < limit));



16

Timed while() loops?

```
float limit; // define variable at start
...
limit = seconds() + 7.5; // set time limit for loop
do
{
// lots of instructions here
}
while( !digital(9) && (checkbeacon(2)==mybeacon)
&& (seconds()<limit) );
```

- when loop ends, don't know why - must check
 - if (digital(9)) // hit something
 - else if (checkbeacon(2)!=mybeacon) // lost beacon
 - else // time limit reached



17