Project Details

Name: Speedway RFID Swimming Prototype
Researcher: Richard McCarthy
Date: 1st May 2020

Work Carried Out

Further analysis and design of an algorithm to solve issues surrounding approximating a swimmers time and lap count. Details of the algorithm that will be developed are given below.

The requirements around this have been narrowed down to a competitive pool environment which has greatly reduced the number of variables to contend with.

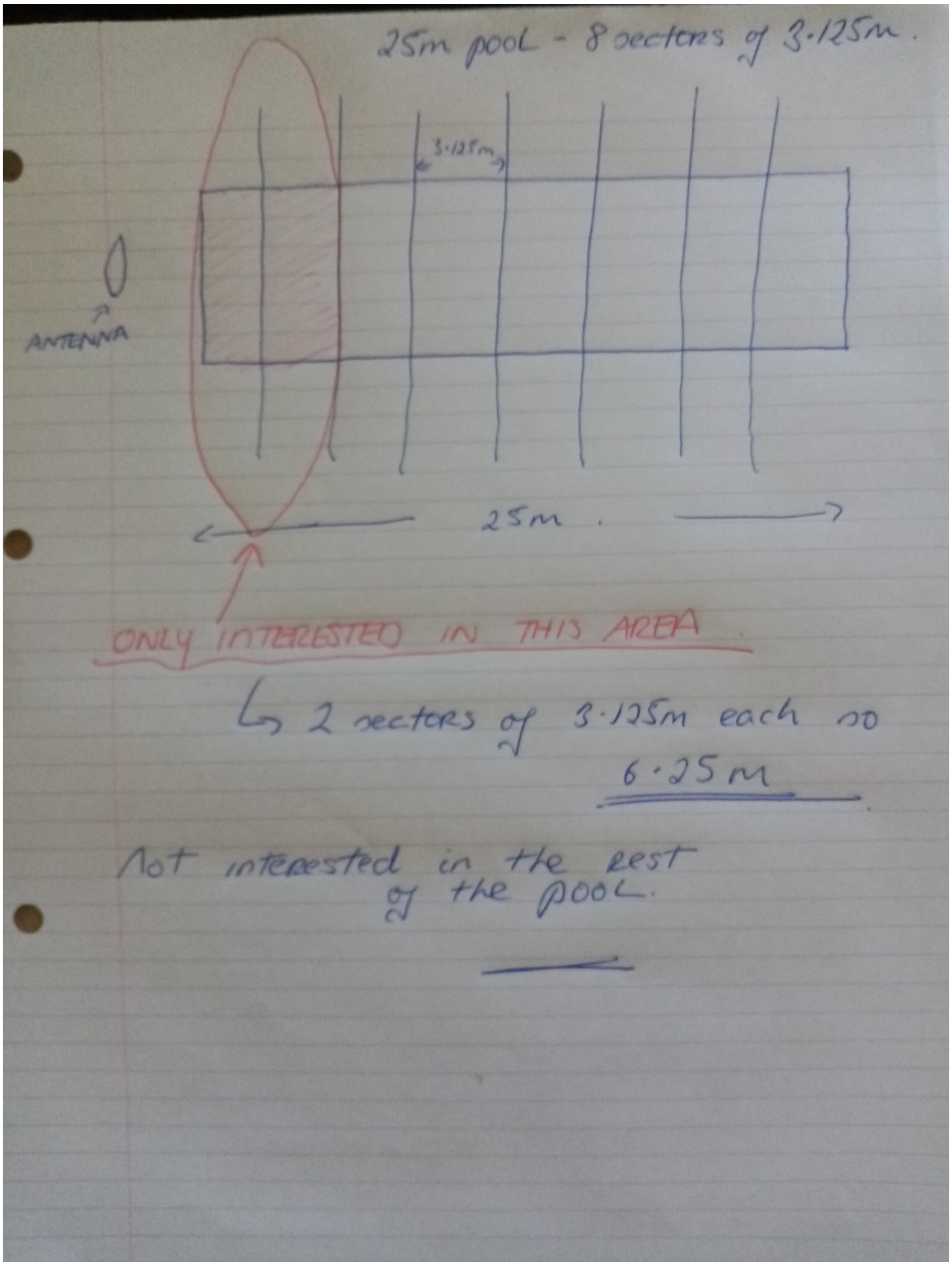
A start has been made on a more complete technical design based around the “Catch-All” algorithm solution.

A proposed solution to the “***Diving Problem***” has also been drafted. This has been provided below. This can be reviewed in the coming weeks once the “Catch-All” solution has been started and progressed.

At that point if the diving solution is deemed workable it can be added into the “Catch-All” algorithm, which is has been designed around. This will be dependent on time and complexity involved.

The Catch-All Solution

From the image below, only interested in 2 sectors. This allows approximate timing to be estimated also which is narrowing down the problem.



This proposed solution is based on 2 variable - RSSI and Read Rate (RR) and is going on a working assumption that when a swimmer is close to the wall (start/stop/turn point) that there will be a grouping of both a High RSSI and a High Read Rate.

This proposed solution is also based around timing and tracking the swimmer from approximating when a swimmer is at the wall.

3 variables -

AtWall = Swimmer at the wall
H2R - High RSSI and High Read Rate

Assumptions for the purposes of this solution

1. 1 swimmer
2. 1 antenna facing down the pool
3. Swimmer starts at the antenna side
4. Swim club squad style training session, so is structured
5. No diving
6. Submerge when push off at the start for each stroke
7. Tag worn top of hat
8. Using Dual-target session for maximum read of the tag
9. Applicable to all strokes
10. High RSSI and High Read Rate (H2R) when swimmer at the wall

Reduction of variables we care about assuming RSSI and RR prove useful.

Point of interest is swimmer at wall - essentially like getting the probability of swimmer AtWall given H2R.

Not dealing with discrete pre-defined values at certain points but the idea is similar.

So Classify and define ranges. High RSSI coupled with High Read Rate - allows it to be reset in code later based on based on pool findings so this is flexible. Plus as it is a classifier, this could be swapped to another metric and use the same algorithm with no changes required.

For example a classifier could be something like an RSSI range from [45 - 60] coupled with a read rate range of [20 - 40] per second.

An algorithm around this can be created based on these assumptions and making certain inferences for now.

NOTE: All timing inferences can be changed later without affecting the structure of the algorithm.

The proposed algorithm is broken up into 3 main parts that once abstracted allow for much more flexibility with adjustments with real data later on.

CLASSIFIER - the range of values for RSSI and Read Rate when swimmer at the wall
ACTIONS - the sequence of main actions involved in swimming a lap in any stroke
STATES - (***Still defining this***) but properties of the actions that are relevant

A LAP IS A SEQUENCE OF MONITORED ACTIONS BASED ON A CLASSIFIER.
This holds for all strokes.

H2R = High RSSI & High Read Rate

ACTIONS

① Standing at wall ⇒ H2R > 3 seconds.

② H2R or lower after 1 to 3 seconds after it was H2R > 3 seconds
⇒ swimmer started.

③ No signal > 10 seconds then get readings
⇒ on return lap.
⇒ expecting a turn or stop within 5 seconds max.

④ H2R or lower after 1 to 3 seconds from having H2R for more than 3 seconds AND was just in previous action ③
⇒ swimmer on new Lap.

→ When in ②, ignore any data until reach ③ action.

• If H2R after back in ③ (returning) goes on for more than 2-3 seconds
⇒ swimmer stopped and back in ①.

Sequence 1 lap {1, 2, 3, 1}

2+ Laps {1, 2, 3, 4, 3, 4, ..., 3, 1}.

↳ This allows abstraction from all the data into a structured form that can approximate time, count Laps, identify swim sets & conclude the session

Proposed Diving Solution

This is been based around certain assumptions for now to simply the process and try and understand it better to create a core solution that potentially could be extended later if proves robust enough.

2 different approaches are provided here.
The 1st is the simpler and potentially easier of the two to implement. This could be an important consideration given timelines.

Assumptions for this proposed solution

1 - Same antenna set up as in the “Catch-All” solution
2 - The antenna is at the same starting side as the dive
3 - Currently just considering 1 dive at the start of the warm-up

<u>Option 1 - Scrub the first two laps</u>

Assuming initial dives only done during the warm up start for a standard squad session.
In a 2500 metre session, if willing to take a loss on the first 2 laps due to the dive then will be covering 98% of the session.

After the swimmer dives, submerges and appears then the swimmer will be out of range of the antenna and on the way down the pool. On the return lap when the swimmer comes into range the algorithm picks up an Action position of 3 (as defined in the “Catch-All” solution).
The algorithm now knows it has a swimmer on a return but its current position is in 1 or “stop/at the wall”. This can be set as rule for algorithm to know a dive happened and it continues as normal once the swimmer is at the wall going from Action 3 to 4 (for turning) and starting the time then but ignoring the first two laps.

<u>Option 2 - Manual intervention</u>

A requirement for this approach would be that each row of swimmers dive from the blocks at a pre-agreed upon interval, e.g. 5 or 10 seconds between each row of swimmers diving.

A person sitting at an application would firstly enter how many rows of swimmers are going to be diving (e.g. 5 swimmers go first, then another 5 then another 2 so that would be 3 rows of swimmers). This person also would press a start button to begin timing for the first row of swimmers. Each subsequent row of swimmers must go at the agreed upon interval.
This manually pressed start time is sent to the system (*this design would need to be figured out but would certainly contain complexity not to be underestimated*) and becomes available to the algorithm either in real-time or afterwards for post-processing of the logged data.

In either case the algorithm would pick up the swimmer on the return lap as described above in *Option 1* and start its timing from the first antenna side wall turn. Now that the system knows it has a manual start time (either in real-time or later for post-processing) it can use the time at the wall turn minus the manual start time to work out the time of the first two laps.

So Diving 50m time = AtWallTurnTime - ManualStartTime

For subsequent rows of swimmers, some assumptions would need to be inferred.
Firstly that each row of swimmers departed at the agreed internal, e.g. 5 seconds.
Secondly once the algorithm picks up swimmers on the first return length it will allow any reading for maybe 5 - 7 seconds that is registered to be assigned the manual start time. After this 5 - 7 seconds it will assume every reading is from the second wave of swimmers and calculating their dive time first two lengths as:

Subsequent row swimmer = AtWallTurnTime - (ManualStartTime + 5)

This would continue for each row of swimmers until the manually entered number of swimmer rows was reached. At this point the algorithm continues as normal in the "Catch-All" solution again.

Something along the lines of the following would track each row of swimmer diving -

Let AgreedInterval = pre-agreed interval the swimmers dive at
Let MaxNumRows = the manually entered number of rows representing swimmers diving
Let ManualStartTime = the manually pressed started time of the first row of swimmers
Let i = current row number of swimmer

```
For i = 1 AND i <= MaxNumRows {
  Initial 50m Diving Time = AtWallTurnTime - ( ManualStartTime  + ( AgreedInterval * i ) )
}
```

Known Blockers

Currently none

Next Steps

Complete technical design of the MVP.
This will be detailed to a level that will be beyond what will be delivered for this phase of the project but will set the ground work on two fronts -

1) Considering a design that will allow the initial building of software that is beyond proof of concept and heading towards a MVP but with the facilities to extend further later and not just be thrown away once verified to allow the “real” product to be built.
2) Provide the foundations for creating a technical roadmap

Once this technical design is completed then building of the software will continue with a much more focused direction given the spec and algorithm solutions agreed on