An Embedded Tracking System Using Sensor Fusion & RTOS

G. Navel and J. Quillès
School of CSE, University of Salford, Manchester M5 4WT, UK
{g.navel, j.quilles}@net.estia.fr

***Abstract* - This assignment consist in implementing a tracking algorithm to track humans with a one axis actuated panel holding two different kind of sensors. The final goal is to prove that using sensor fusion can improve the tracking. In order to prove that we focus on implementing three different algorithm, two simple tracking algorithm, one for each kind of sensor, and one using the fusion of both sensor. Then we compared the results of the three different algorithms two by two using T-Test in order to try to prove statistically that one algorithm is quantitatively better than others.**

Index Terms – Tracking System, Sensor fusion, Statistics.

I. INTRODUCTION

Nowadays robot are more and more integrated in our world. To make them cleverer and facilitate their integration, it can be useful to ease their interactions with humans. This assignment consist in implementing a tracking algorithm able to track humans. Indeed, have the ability to "look at human users" or "follow them" can be useful in a lot of different kind of interaction with human. Surveillance, telepresence and assistance are different fields where it could be use.

The material given to us, consist in a one axis panel actuated by a servo motor. This panel was holding two infrared telemeter (sharp sensors) and a 4*4 pixel thermal camera. It was ask to us to implement a RTOS in an atmega128 with real task handling, a bidirectional serial communication between the microcontroller and a PC and a JAVA GUI (with intelliJ) on a PC to control the panel orientation.

The basics knowledge requirements to complete this assignment were C and Java programming, AVR microcontroller register configuration, IntelliJ GUI development, Sensor fusion implementation, RTOS implementation, Statistics knowledge, Protocol communication implementation serial and I2C, Image treatment, and technical knowledge about components used.

A. Literature Review

Here is a summarized review of the papers and lectures used to complete this assignment:

"Algorithmique et Programation" Lectures about how to program in C and think about algorithm implementation by Guillaume Riviere, ESTIA first year. It include all the basics of the C language and the good habit to take when programming.

"Initiation aux Statistiques appliquées" Lectures about how to use statistic to prove results, by Virginie Rosa with the collaboration of Audrey Abi Akle, Nadine Couture, Katarina Borgiel and Marion Saumonneau, ESTIA, second year.

"Programmation Orienté objet" Lectures about learning how to program in Java by Sebastien Bottecchia and Guillaume Riviere, ESTIA second year.

"Capteurs et communication" Lectures about how to implement communication protocol via UART by Joseph Canou and Olivier Patrouix, ESTIA, second and third year.

"Signal et Image" Lectures about how to treat pictures by Sebastien Bottecchia, ESTIA, third year.

"Embedded system & RTOS" Lectures dealing with, C programming, AVR microcontroller, and RTOS implementation by Theodoros Theodoridis, University of Salford, embedded system master.

"Mechatronics & embedded robotics" Lectures with some parts dealing with sensors fusion and statistics, by Theodoros Theodoridis, University of Salford, embedded system master.

B. Paper Outline

The rest of the paper is organized as follows: Section II presents the methods employed with subsection A. describing the hardware architecture, subsection B. describing the software architecture and subsection C explaining algorithms employed. Then, experimental results and analysis of the performance of the system are demonstrated in subsection A. and B. of the Section III. Finally, conclusions and future directions are given in Section IV followed by acknowledgments and references.

II. METHODS

A. *Hardware architecture*

(a) *Actuated panel and sensors*

The tracking panel used for this assignment was a one axis motorized panel actuated by a standard S3003 servo motor. The motorized axis is arbitrary chosen as the z axis. "teta" is chosen as the angle between axis x0 (fixed with the body of the servo motor) and x1 attached to the moving panel as shown below:

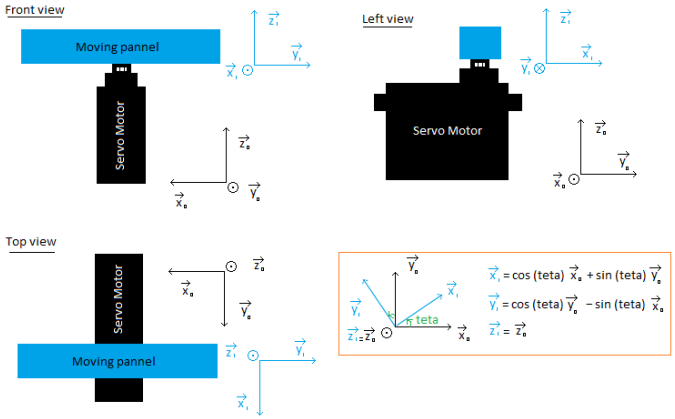

Figure 1. Geometrical considerations.

The angle teta can vary from 0 to 180° (Physically tested) and on the moving panel there are two IR Sharp range sensor and one thermal Omron camera disposed as shown below:

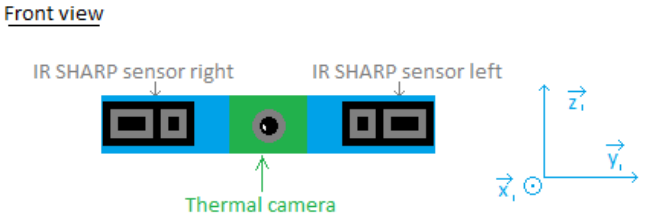

Figure 2. Sensors disposition

The IR Sharp GP2Y0A02YK range sensors are analog sensors specified to measure distance from 20cm to 150cm.

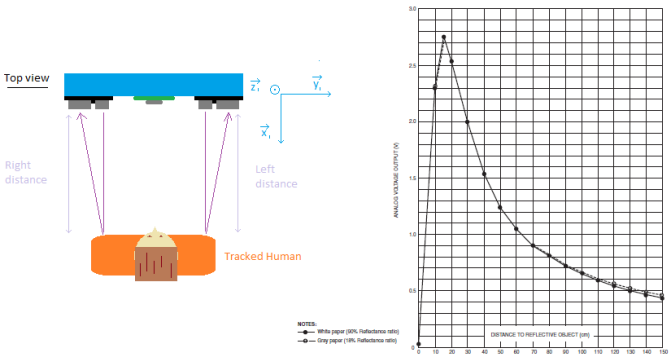

Figure 3. Distance measured by IR sharp sensors (img ref [13] p4)

The thermal Omron Camera is the model D6T -44L -06. It's a 4*4 pixel camera sending the 16 thermal value 2 bytes long of each pixels plus the value of the ambient temperature via I2C.

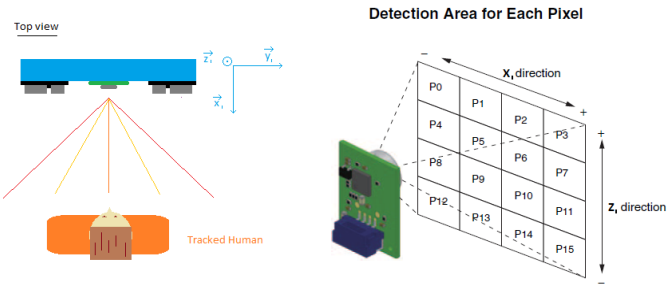

Figure 4. Thermal sensing by Omron camera (img ref [9] p2)

(b) *Control units*

To interface with the servo and the sensors we used a STK300 Kanda development board. This board contain the Power Supply Unit (PSU) based on a LM2576, the USB Control Unit (UCU) based on a FT232RL, and the Microcontroller Control Unit (MCU) based on an ATMega128. The computing process is deported on a separate computer with a Java GUI interface.

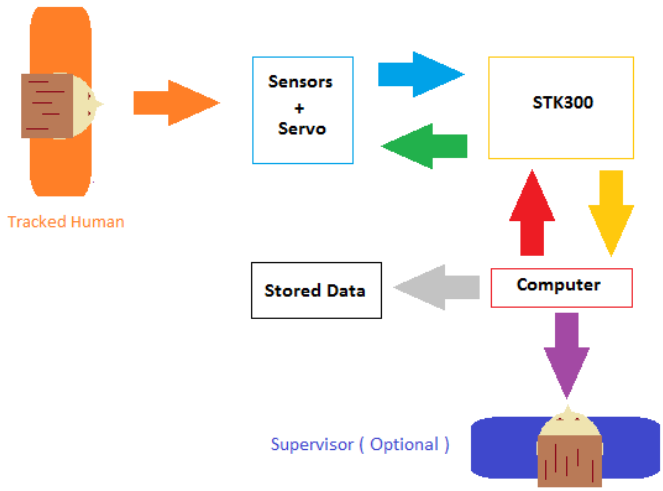

Figure 5. Flux, data and control

- Sensors get data from the human tracked target. (orange)
- STK300 read sensors data (blue) and send it to the computer via USB Serial communication (gold). It also receive the high level servo position command from the computer (red) and generate the according PWM to the servo motor (green).
- The computer received sensors data (gold), compute it to elaborate a high level servo position command and send it to the STK300. (red) Then the computer store the data into an excel file. (grey) The computer also display received data (purple) to the optional supervisor.

B. *Software architecture*

(a) *Microcontroller level*

The ATMega128 have been programed in C language with the Atmel Studio IDE. The YAVRTOS, a free and open source, real-time task handling operating system for AVR microcontroller, is implemented on the ATMega128, handling 3 tasks which are, sensors acquisitions, serial communication to send acquisitions to the computer, and read and execute command sent by the computer. The task with the highest priority is the sensor acquisition and the task with the lowest priority is the reading and execute command from the computer. The code is organized in package with each package dedicated to kind of functions.

- "adc" contain all the functions relative to analog to digital conversions, useful to get data from analog sensors.
- "camtwi" contain all the functions relative to communicate with the thermal camera via I2C and get the pixels thermal value.
- "lcd" contain all the functions relative to display things on a LCD screen, used for debugging mode.
- "port" contain all the functions relative to access and set state port of the ATMega128.
- "rtos" contain all the necessary to implement YAVRTOS.
- "serial" contain all the advanced function we implemented to encode the serial communication. Data are encoded with start and stop bytes, specials characters consideration and checksum calculation.
- "servo" contain functions to control the servo motor. is implemented
- "twi" contain all the basic functions relative to standard I2C communication
- "uart" contain all the basic functions relative to standard serial communication
- "main" contain tasks definition and launch the YARTOS and the tasks handling.

(b) *Java GUI level*

We implemented a classic MVC structure (Model, View, and Controller) to organize the Java code on the computer.

The final implemented view look like this:

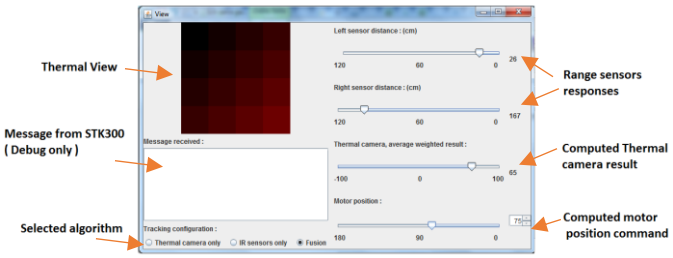

Figure 6. Final GUI Java view implemented

Due to the three different tracking algorithm we want to use, we choose to implement multiple model file, one for each algorithm, plus one to store value into the excel file.

The resulting project organization:

- "View" is the class displaying the window interface.
- "Model" is the class writing data into the excel file
- "CameraModel" "SensorModel" "FusionModel" are the models for each algorithm and motor position calculation.
- "SerialPort" is the class who write and read data to and from the SerialPort.
- "SerialTreatment" is the class used to interpret the advanced serial protocol we implemented
- "Controller" is the class linking view, models and serials classes
- "Run" creating the controller object to launch the app.

C. *Algorithms employed*

Algorithms employed are based on the sum of current position (belonging 0° to 180°) and an increment (or decrement) moving value calculated from the sensors (belonging -6° +6°). If the sum is below 0 then the commanded position is set to 0°, if it's above 180 it is set to 180°, else it is set directly to the sum.

$$0 \leq New\ command\ position = Curent\ position + Value \leq 180 \quad (1)$$

(a) *Sharp only "Value" calculation*

With Sharp sensors only, the 'Value' to move (SV), in degrees, is calculated from the difference of the right and left distances measured by the two sensor, regarding the value MinRL= Min (right, left), in cm, to take in consideration if the tracked target is close to the panel. If the target is near the panel the effect we choose is to move more the panel than if the target is far.

TABLE 1
SV CALCULATION TABLE

MinRL (cm) / ***Distance subtraction (cm)***	***MinRL ≥ 40 (Target far)***	***MinRL < 40 (Target near)***
Right-Left > 60	3	4
60≥ Right-Left > 30	2	3
30≥ Right-Left > 10	1	2
10≥ Right-Left ≥ -10	0	0
-10> Right-Left ≥ -30	-1	-2
-30> Right-Left ≥ -60	-2	-3
-60> Right-Left	-3	-4

(b) *Thermal camera "Value" calculation*

With the thermal Camera only, the 'Value' to move (CV), in degrees, is calculated from two parameters.
First, WAV: the Weighted Average thermal Value of the pixels P[i][j], a 10 bit signed value.

$$WAV = \frac{\Sigma_{j\in[0;3]}\left(a[j]*\left(\Sigma_{i\in[0;3]}\frac{(P[i][j]-Pmin)}{(Pmax-Pmin)+1}\right)\right)*1023}{\Sigma_{j\in[0;3]}\ |a[j]|} \quad \begin{matrix} With: & a[0]=-2 \\ & a[1]=-1 \\ & a[2]=1 \\ & a[3]=2 \end{matrix} \quad (2)$$

Note: WAV can belong [-511; +511] in theory with very particular situation, but by experimentation it stays between [-100; 100].

Second parameter: TAV: the Thermal Average Value of the pixels P[i][j].

$$TAV = \frac{\sum_{j\in[0;3]}\left(\left(\sum_{i\in[0;3]}\frac{(P[i][j]-Pmin)}{(Pmax-Pmin)+1}\right)\right)*1023}{4} \quad (3)$$

WAV indicate if we should move a lot or not and TAV indicates if tracked target is near or far.

TABLE 2
CV CALCULATION TABLE

AWV \ TAV	*TAV ≥ 600 (Target near)*	*TAV < 600 (Target far)*
AWV > 90	3	4
90≥ AWV > 60	2	3
60≥ AWV > 30	1	2
30≥ AWV ≥ -30	0	0
-30> AWV ≥ -60	-1	-2
-60> AWV ≥ -90	-2	-3
-90> AWV	-3	-4

(c) *Fusion Algorithm*

The fusion is the simple crossing of Camera to move Value CV and Sharp to move Value SV, plus a truth parameter for each. The Fusion 'Value' to move (FV), in degrees, is calculated with CV and SV: FV = fusion (CV, SV)

TABLE 3
FV CALCULATION TABLE

CV \ SV	-4	-3	-2	-1	0	1	2	3	4	?
-4	-4	-4	-3	-3	-2	-2	-1	-1	0	-4
-3	-4	-3	-3	-2	-2	-1	-1	0	1	-3
-2	-3	-3	-2	-2	-1	-1	0	1	1	-2
-1	-3	-2	-2	-1	-1	0	1	1	2	-1
0	-2	-2	-1	-1	0	1	1	2	2	0
1	-2	-1	-1	0	1	1	2	2	3	1
2	-1	-1	0	1	1	2	2	3	3	2
3	-1	0	1	1	2	2	3	3	4	3
4	0	1	1	2	2	3	3	4	4	4
?	-4	-3	-2	-1	0	1	2	3	4	0

CV =? if no thermal object detected. (Max < Ambient)
SV =? if no object detected or object too far (MinRL>120).

III. RESULTS

A. *Experimentation*

To compare our algorithm we tested them by making them track a walking person wandering in a circle around the tracking panel, from 0° to 180° and then from 180° to 0°, at an approximatively constant walking speed and at an approximatively constant distance R≈ 1m from the panel.

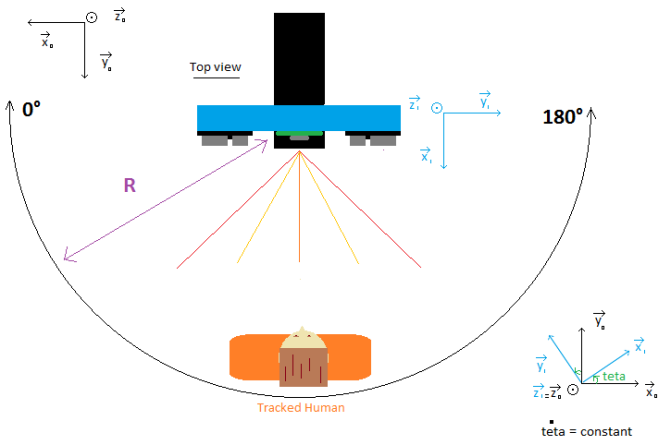

Figure 13. Tests configuration

For each algorithm we execute 20 times the tracking test and we qualitatively describe the tracking and calculate the average error e between teta_r, the successive real angular panel positions obtained by tracking, and teta_w the angular position wanted each time.

$$e = \sqrt{\frac{1}{n} * \sum_k (teta\_r - teta\_w)^2} \quad (4)$$

TABLE 1
QUALITATIVE OBSERVATIONS TABLE

Characteristics	*Sharp*	*Camera*	*Fusion*
Fluidity	Not Soft	Soft	Soft
Fast	Faster	Slower	Medium
Track all objects	Yes	Only hot	Yes
Track objects far	No	If hot	If hot
Tracking	Good	Good	Good
Stability	Not stable	Stable	Stable
Error (average)	6,05	8,4	6,4

TABLE 2
QUALITATIVE OBSERVATIONS TABLE

Type	*Sharp*	*Camera*	*Fusion*
Population	20	20	20
Error (average)	6,05	8,4	6,4
Variance	3,1475	1,54	2,34
Standard deviation	1.82	1,27321	1,56945
Standard error	0.407	0,2847	0,35094
Min	4	5	4
Max	11	10	9
Range	7	5	5
Median	6	8,5	6
Max of 95%	6,83	8,94	7,07
Min of 95%	5,27	7,86	5,73
Confidence interval	1,56	1,08	1,34

Distributed result population for each experiments.

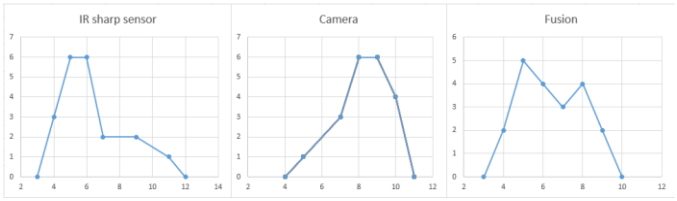

Figure 15. Distributed result population for each experiments

B. *Analysis*

It's hard to qualitatively find which tracking algorithm is better than others but it seems that qualitatively the fusion tracking is very similar to the camera tracking …
However we can consider that camera and sharp are normally distributed. More, even if fusion distribution is not looking like normally distributed we assume that fusion is also normally distributed to allow us to perform a T-Test, but we need to do more experiments to verify it.
In order to reveal if our fusion tracking is significantly different from using just each sensor separately we choose to perform T-Tests, between (Fusion, Camera) and (Fusion, Sharp) but first we verified that tracking using camera and sharp only are significantly different using a T-Test.

- T-Test Sharp/camera results: t(38)=4.73, p=1.5E-05<0.05 ≠H0.

This result allow to conclude that camera tracking and sharp tracking are significantly different.

- T-Test Sharp/fusion results: t(38)=0.65, p=0.2593>0. 05 =H0.

This result can't allow us to conclude that fusion tracking and sharp tracking are quantitatively different.

- T-Test Camera fusion results: t(38)=4.426, p=3.9E-5<0.05 ≠H0.

This result allow to conclude that camera tracking and fusion tracking are significantly different.

IV. Conclusions

To conclude about results obtained, first we can say that fusion algorithm and sharp algorithm are quantitatively significantly different from the camera algorithm and by this way significantly better than the camera algorithm, because they are different and both with a better mean.

Second, even if the quantitative analysis seems to show that fusion tracking algorithm doesn't bring significant quantitative amelioration in comparison with the Sharp algorithm, we can affirm that using the fusion is an amelioration because fusion have qualitative improvement. The system with fusion algorithm can now track far object if hot. It is stable and the movement is softer like the tracking with camera only.

However to verify our conclusion we should experiment more times because 20 for each experiment is not a big population. More to improve results it could be interesting to try different values for simple sensors algorithms and try other kind of fusion models such like TLU and so…

Then in order to try to determine optimal algorithms, more experiments should be done with tracked target at different distances from the panel and different "walking speed". Moreover, if working on this project last in the time, it should be interesting to create a complete testing system with an actuated thermal moving target with parametric angular speed and distance "to target" easily modifiable. This could allow to realize experiments faster and more efficiently than by tracking a real person walking at an unknown walking speed and give better tests result.

Acknowledgments

Thanks to ESTIA and University of Salford for the knowledge and materials given.

Thanks to Dr. Theodoridis for, giving this very interesting project, and guide us during those five weeks even if it was not healthily easy for him.

Thanks to my family who support and follow me in all my projects.

References

[1] G. Riviere, *Algorithmique et Programmation*, ESTIA, lectures 1st year.
[2] V. Rosa, A. Abi Akle, N. Couture, K. Borgiel and M. Saumonneau *Initiation Statistiques appliquées* ESTIA, lectures 2nd year.
[3] S. Bottecchia and G. Riviere *Programmation Orienté objet*, ESTIA, lectures 2nd year.
[4] J. Canou and O. Patrouix, *Capteurs et communication*, ESTIA, lectures 3rd year.
[5] S. Bottecchia, *Signal et Image*, ESTIA, lectures 3rd year.
[6] T. Theodoridis, *Embedded system & RTOS*, University of Salford, lectures from embedded system master.
[7] T. Theodoridis, *Mechatronics & embedded robotics*, University of Salford, lectures from embedded system master.
[8] T. Theodoridis, *IEEE Assignment Template*, University of Salford, Standard template.
[9] OMRON corporation, *D6T MEMS Thermal Sensors*, Datasheet.
[10] OMRON corporation, *[D6T-44L -06] Application Note No.MDMK-12-0493*, Datasheet.
[11] Atmel corporation, *ATMega128*, Datasheet.
[12] Kanda Systems, *STK300 User Manual*, User manual.
[13] SHARP, *GP2Y0A02YK Optoelectronic Device*, Datasheet.

External links for online statistic calculations

- http://www.usablestats.com/calcs/2samplet
- http://www.socscistatistics.com/tests/studentttest/
- http://studentsttest.com/

Article written by QUILLES Jonathan alias Mike118.