Finger Gesture Sensor (FGS), an open-source input interaction technology

by Frank Cohen, part of the Reflections project
https://github.com/frankcohen/ReflectionsOS
February 18, 2022

Why is this needed?

Improvements in user experiences come from moving beyond buttons and the mouse. Finger gestures with haptic feedback allow for more sophisticated interaction with information systems.

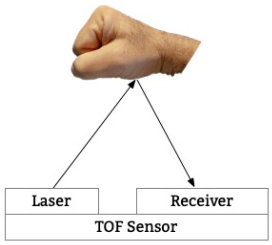
Finger Gesture Sensor (FGS) is an open-source hardware, software, and Internet architecture. It uses advancements in Time Of Flight (TOF) sensor hardware with a pattern recognition engine. FGS scales from user interaction with a wrist watch up to large industrial controls.

The sensor and architecture

Time of Flight (TOF) sensors identify the distance between the sensor and an object. They are useful for drones needing to detect the distance to the ground during a landing. They work by emitting a laser and sensing the time it takes for a reflection to return to the sensor's receptor. When we package a grid of TOF sensors together one application is finger gesture sensing.

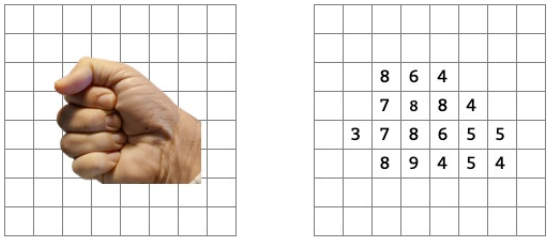
The algorithm

A single Time of Flight (TOF) sensor identifies the distance to an object. For example, consider a hand making the letter 'A' symbol in American Sign Language (ASL). A side view of a hand being sensed by a single TOF sensor.



With a single sensor the system detects the hand's presence. It does not know enough to detect the hand in an ASL pattern for the letter 'A'.

New TOF sensors detect multiple objects in a grid. The following shows a TOF sensor detecting multiple distance readings of the front of the hand. The readings identify the distance to each part of the hand:



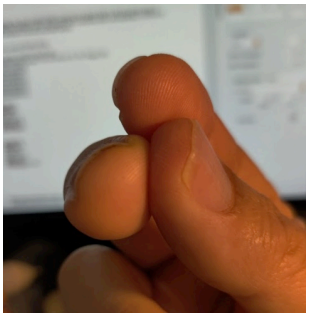
		8	6	4			
		7	8	8	4		
	3	7	8	6	5	5	
		8	9	4	5	4	

Finger Gesture Sensor (FGS) learns the pattern by recording an 8x8 grid of measurements. For example, the thumb in the above image is slightly further away from the sensor than the pinky finger.

FGS stores the distance readings in a binary pattern file. The pattern files end in “ges”.

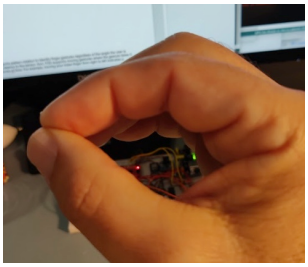
FGS applies two sets of filters when matching the pattern files to the live sensor data: it compares the distance each cell reports with a sensitivity filter, and identifies a threshold of overall matches to the pre-recorded pattern.

Matching the live TOF data to the recorded patterns lets us identify finger gestures. For example, when making this “pinch” formation with our fingers, FGS detects a “Hit” with the Pinch3.ges recorded pattern.



```
/left.ges 23
/left2.ges 15
/Oh.ges 9
/SignA.ges 21
/Right.ges 4
/Pinch3.ges 27 HIT
```

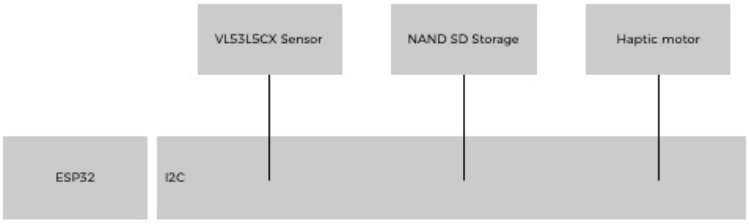
Next to each pattern file is a count of measurement matches. Pinch3.ges surpassed the threshold of measurements for FGS to identify as a “HIT”.



```
/left.ges 0
/left2.ges 0
/Oh.ges 27 HIT
/SignA.ges 1
/Right.ges 1
/Pinch3.ges 6
```

Implementation details

A reference implementation uses an ESP32 processor, VL53L5CX sensor, SD flash storage, and haptic motor. I2C bus connects the devices. The application does not require high speed communication between devices and the VL53L5CX sensor package comes with I2C capabilities built-in.



The implementation uses the Spark Fun VL53L5CX breakout board and Spark Fun provided software driver for Arduino IDE environments.

Sensor details at:
VL53L5CX, the data sheet is at https://www.st.com/resource/en/datasheet/vl53l5cx.pdf
SparkFun Electronics VL53L5CX library

https://learn.sparkfun.com/tutorials/qwiic-tof-imager---vl53l5cx-hookup-guide

The implementation is an Arduino IDE sketch. The sketch builds gesture learning files (.ges) and compares them to the live readings from the VL53L5CX sensor. Here is a summary:

a) Read the existing gesture patterns into a memory buffer. Each is an 8x8 array of distances from the VL53L5CX.

b) At the user's option take a live reading from the sensor and save the 8x8 array of distances to a gesture file. Gesture files are binary data streaming the 64 readings from the sensor (8 cells times 8 rows).

c) Identify % similarity between gesture patterns and the live sensor data. Return an integer value of measurement similarities. Identify the pattern that achieves a threshold as a "HIT".

Find the implemented software at https://github.com/frankcohen/ReflectionsOS/tree/main/Devices/Sox/Software/14Gestures

New feature needs

The reference implementation does not provide these features:

1) Support finger gesture sensing where the fingers may rotate 360 degrees around the gesture pattern. Supports this by iterating the gesture pattern by 22 degrees of rotation, comparing to the live sensor data, then repeating the 22 degrees rotation and comparison 16 more times.

2) Allow the gesture pattern to signal a haptic feedback motor to indicate a HIT. Each pattern may have its own haptic feedback style.

3) Enable multiple second gestures. Expands the gesture pattern files to save 6 patterns over a 3 second time period. And does the check against the most recent 3 seconds of live sensor data.

Limitations

The architecture does not address these issues:

1) Will the design support people with little or big fingers?
2) How close to the sensor should be the focal length?
3) How to discern between gesture profiles that generate multiple "hits"?
4) Is there a difference in sensing gestures between indoor and outdoor environments?

How to participate and use the technology

Finger Gesture Sensor is a free open-source software. FGS is licensed under a GPL v3 Open Source Software license. Read the license in the license.txt file that comes with this code for details.

FGS is hosted in the Reflections source code repository:
https://github.com/frankcohen/ReflectionsOS