Documentation

Project Name: Automated Turret
Team Name: TechFire
Team ID: 64
Members: **Srivatsan Sridhar**
Arpit Singh
Prateek Kaul

Introduction

The project is about making a turret capable of automatically scanning for a pre-specified target in its vicinity and shooting it. The turret will consist of a gun and a camera mounted on a platform which can rotate in the horizontal and vertical axes. The video feed from the camera would be processed using OpenCV (image processing) and 2 motors will rotate the assembly so as to point at the target. Initial work will be done using a laser pointer instead of a gun. In the second stage the gun will be added and the laser will assist the gun to take aim.

Project Details

Components Used

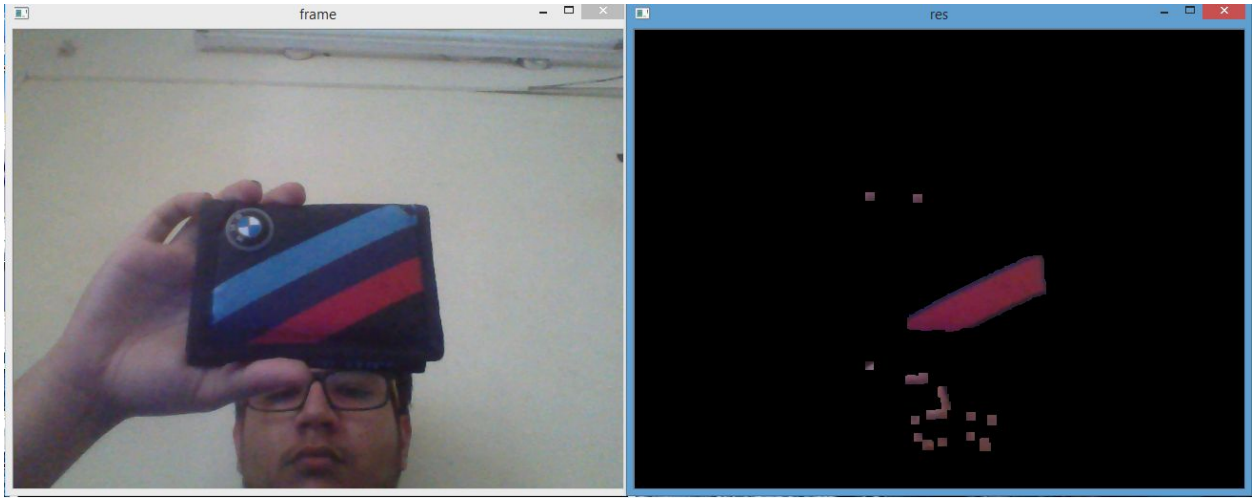
- Arduino Uno
- 6V battery (4500 mAh)
- Nerf N Strike Elite Firestrike Blaster
- Laser pointer
- 3 LR44 cells for the laser pointer
- 3 servo motors: 9kg-cm for lowest platform, 15 kg-cm for middle platform, 3 kg-cm for gun trigger
- L-clamps for fixing the servos
- Wood for making the mount
- Screws, cable ties (zipties), rubber bands, tape

Total project cost: Rs. 5403

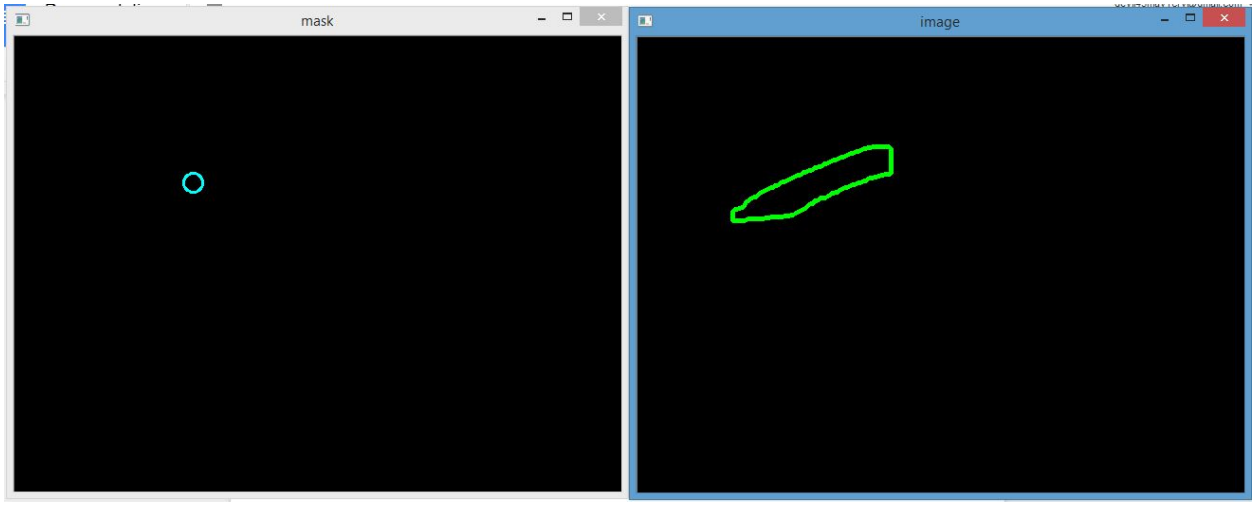
OpenCV

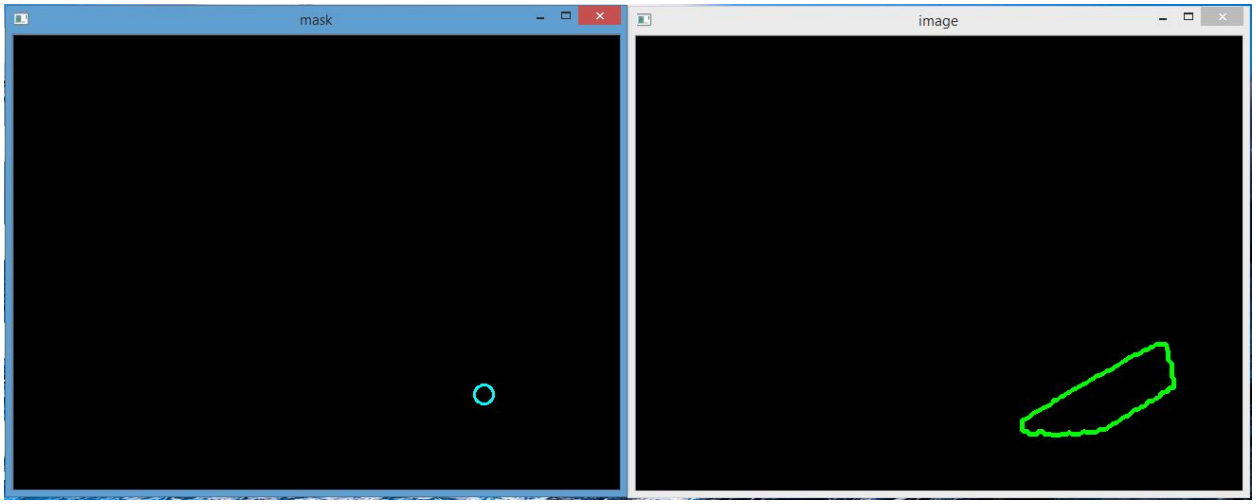
We are using OpenCV (an image processing software) on python for our project. We learned it during the first week and wrote a code to trace the position of an object of a particular colour. The code involves converting the image to an HSV image. Then we scan the image for objects which lie in the same colour range as our object. To reduce the noise we used Gaussian Blur and Dilation techniques. We found that reducing the kernel size in dilation and increasing the number of iterations was more effective. We then find the biggest object of the given colour, assuming that our object would be that object and then create a circle at its centroid. For ease of changing the pre-specified colour, we added setMouseCallback function to change the colour to the colour of the object which is clicked on.

The colour is set to the red on my wallet. The image on the left is the original and the image on the right contains all objects having the same colour. The small blocks are the patches on my face which lie in the same colour range. We find the largest object by drawing contours of the image on the right and only print the largest contour.



The image on the right is the boundary of the largest object and the image on the left is a circle with its centre on the boundary's centroid.





Note that there is still a bit of noise which makes the boundary of the object appear jagged. We are still trying different filtering techniques to reduce it.

UPDATE: Noise considerably reduced by using the Morphological transformation, opening and increasing the iterations of dilation to 5.

Modified OpenCV code with PySerial

The program uses the initial code to find the center of an object of a pre-defined colour**,** then it finds out the deviation of the center of the object from the center of the image, and then sends W,A,S or D to the Arduino via PySerial depending on how to rotate the servo motors.
Example: Say (cx,cy) be the centre of the object and (320,240) be the centre of the image. So if cx-320 > 20, then it sends ‘D’ via the serial port.

Arduino Code

The Arduino code reads the character on the serial port and then responds as follows:-
If the character is:

1. ‘W’, it moves the upper servo by 1 degree upwards.
2. ‘S’, it moves the upper servo by 1 degree downwards.
3. ‘A’, it moves the lower servo by 1 degree leftwards.
4. ‘D’, it moves the lower servo by 1 degree rightwards.
5. ‘K’, it moves the servo attached to trigger the gun, to shoot the bullet

The combination of the above two programs allowed the mount to move in such a way that the centre of the camera came within a 20 pixel distance from the centre of the object.

Later, it was decided to mount the camera on the lowest platform and keep it fixed, because the viewing range of the camera would anyway be the same, and keeping the camera fixed will give a more stable image and make processing easier. Also as we were going to move the laser

pointer to the object, we would need to keep the camera stationary so as to not change the coordinates of the object

Connections required:

Each servo has three wires, for 6V, ground and control. The colour code is red for 6V, brown for ground and orange for control on the servos for movement. For the triggering servo, the code is red for 6V, black for ground and white for control. The control wires for the horizontal motion servo, vertical motion servo and trigger servo are connected to pins 9,10,11 of Arduino respectively. The 6V and ground wires are connected to the respective terminals on the 6V battery. The ground pin of Arduino should also be connected to the negative terminal of the battery.

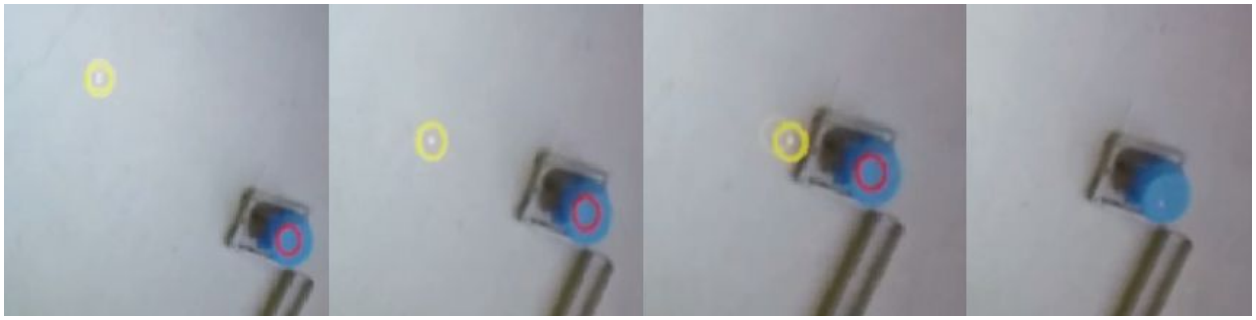
Tracking the Laser pointer:
If we were going to move the laser point to the object, we would need its coordinates. Finding its coordinates employed a similar strategy to finding the coordinates of the object. The only thing that was different was that we needed a much smaller range for the laser pointer as a larger range in the HSV values lead to the inclusion of any white object in the image.

Moving the Laser pointer
Once we had the coordinates of the laser pointer, we swapped the center coordinates(320,240) with coordinates of the object and coordinates of the object with the laser point coordinates in our OpenCV code.

Code in Action:

To our eyes:

Picture showing circles around the centres of the object and the laser point

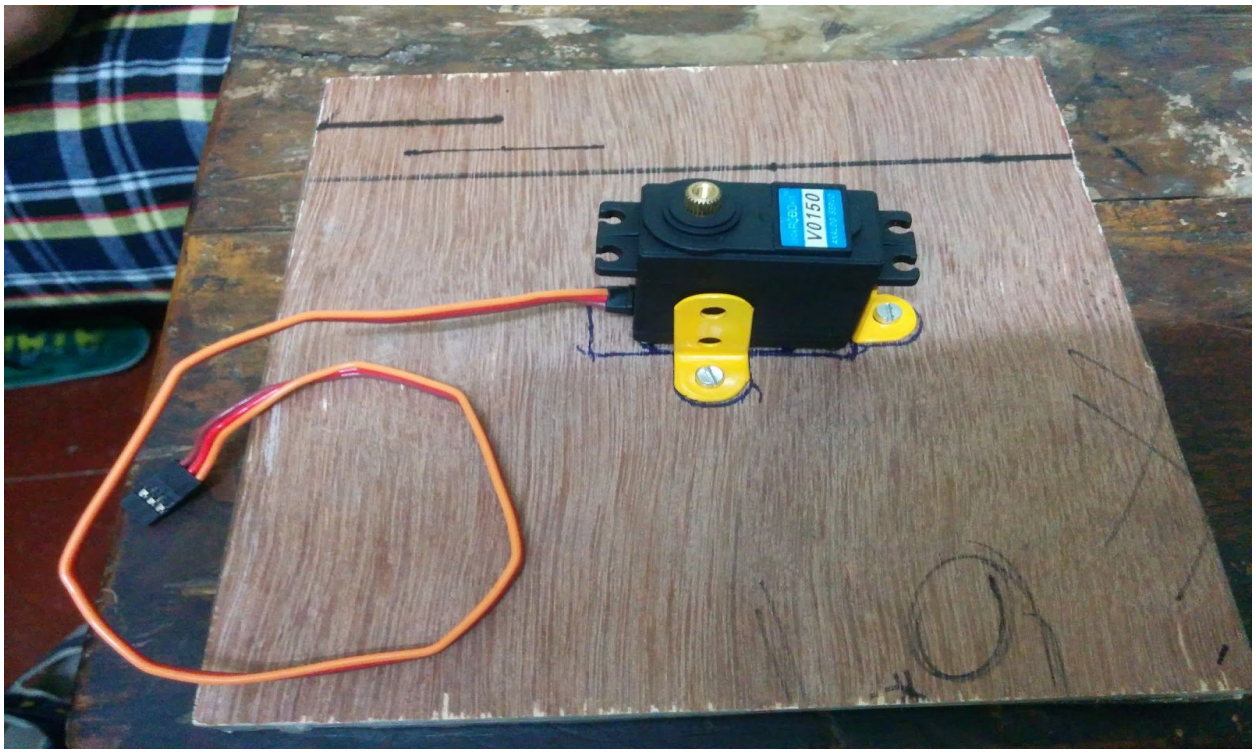


Mechanical Setup

There are three layers to the mount. The bottom layer is fixed and one servo is mounted on top of it. This servo can rotate the middle layer about the vertical axis. Another servo is mounted on the middle layer which can rotate the top layer about a horizontal axis. An L-clamp connects this servo to the top layer. The camera and laser pointer/gun are to be mounted on the top layer.

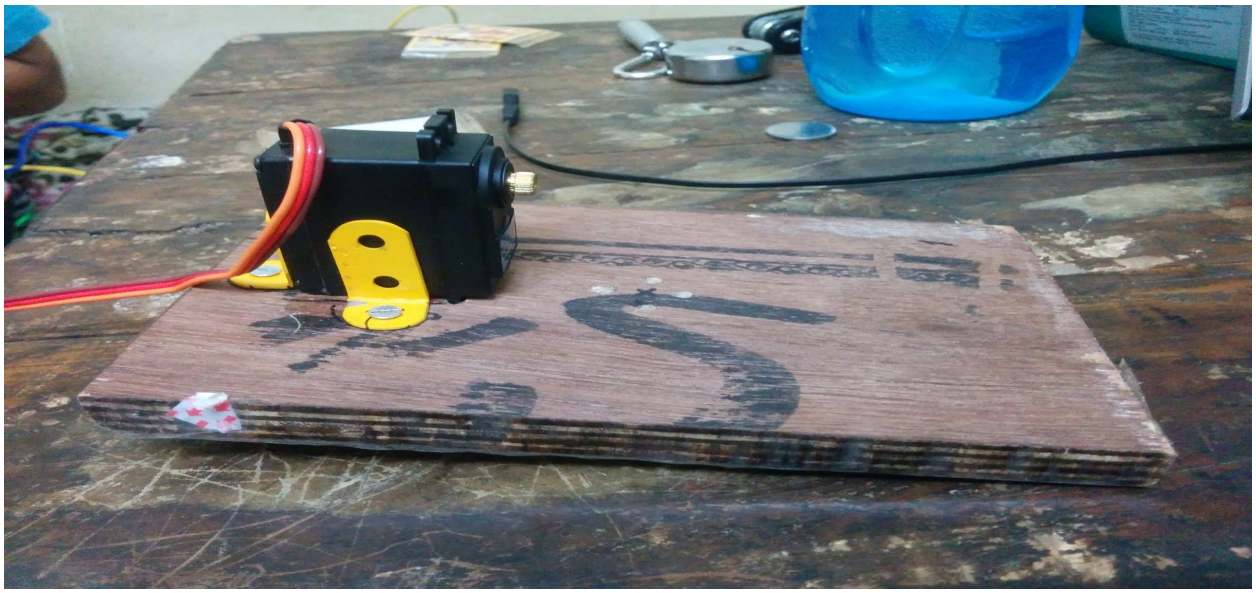
Photos of the mount

Bottom Layer: The servo is held in place with a combination of 4 L-clamps



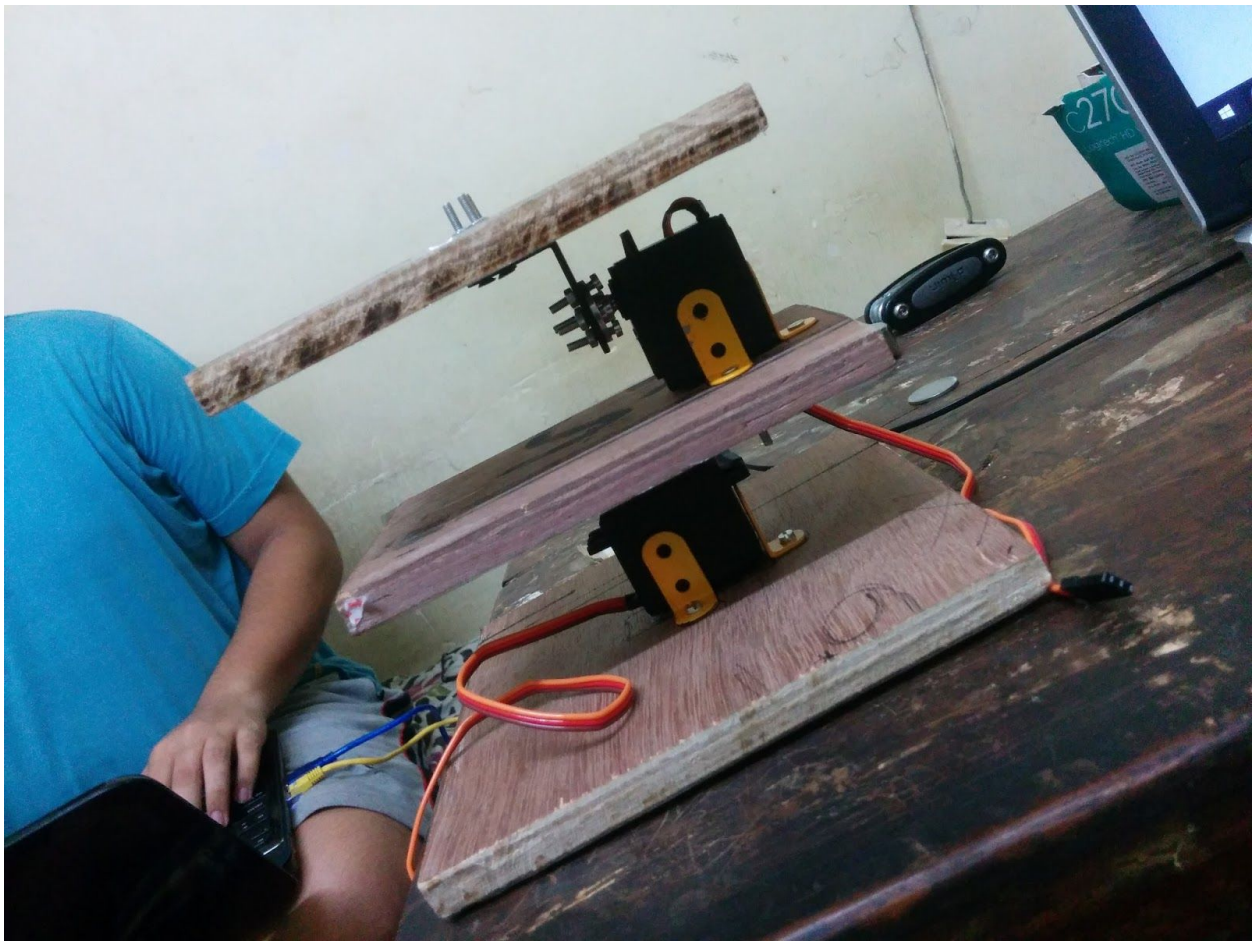
Middle Layer: Similar to the bottom layer, the servo is held in place with 3 L-clamps. Also we

had to make a slot in the platform to fit the part which has been highlighted in the picture above

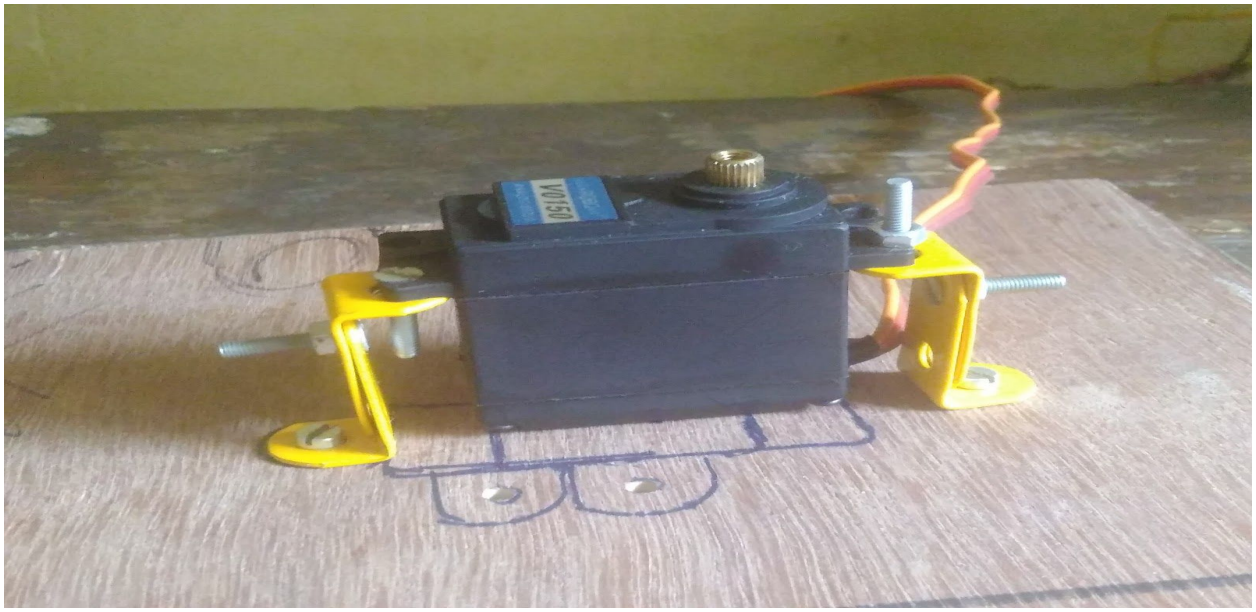


Complete: The base layer servo was attached to the middle layer platform using a metal horn. The middle layer servo was attached to the topmost platform using a combination of the metal horn and an L shaped bracket.


© Robot Gear



We discovered that the bottom servo could be easily toppled so we decided to change the mount to make it more stable. The image below shows the new mount of the bottom servo



Final mount:

The laser pointer was mounted on the topmost platform, using a ziptie passing through two holes. Being on the top platform, the laser pointer can rotate.

Building a toggle switch for the laser: The switch given on the laser keeps it on only as long as the switch is pressed. To overcome this, we made a toggle switch that we learnt from youtube (https://www.youtube.com/watch?v=aswWXFbcedM).
We tied two rubber bands on either side of the switch and tied a zip tie over the switch. When the zip tie is rotated so that the knot is over the switch, it is kept off. When the rope part comes over the switch, it is kept on. Coincidentally, the rubber bands and ziptie formed colours of the Indian flag.


Laser switched off


Laser switched on

Fixing of the Gun

We bought a 3rd servo motor to trigger our gun.

The gun we are using is a Nerf N Strike Elite Firestrike Blaster

The gun was fixed using cable ties and later reinforced by double-sided tape to avoid small movements. All other servos were also fixed firmly using double sided tape. The laser is also mounted atop the gun using cable ties.

The gun was mounted on the top platform and the platform was extended slightly to make space for the servo. The shaft of the servo was connected to the trigger of the gun using a piece of thick metal wire.

Offset of Laser

The laser has a particular offset with the shaft of the gun. Its is approximately 2.8 cm above and to the side of the gun shaft.

For calculating the laser offset in pixels, we used triangle similarity which goes like this:-
We have an object of a known width W, at a distance D from the camera, then the perceived width in pixels of the object is P = FxW/D where F is the perceived focal length in pixels.
The idea was that we know W=2.8cm as the true offset distance and we would convert it to pixels using the above formula.
For using the above formula we needed F and D. F would be a constant but D would keep changing.

Code to find Perceived width of an object:
Our previous code found the largest contour of an object of a particular colour. We made a minimum area rectangle around it and as the object was a circle, the rectangle was a horizontal square. We then calculated the average of its two sides and named it P = perceived width of the object

Calculation of F:
For calculating F, we ran the above code keeping the object(of diameter 4.6 cm) at a fixed distance(83 cm), its perceived width came out to be 52.5 pixels.
So for this case:
W = 4.6 cm
P = 52.5 px
D = 83 cm
So using the formula F = PxD/W we get F = 947.3 px which would be constant for all distances(D).

Calculation of D:
As we now know the value of F, we can calculate D for all cases. Our code calculates P for the object of a known dimension. So D = FxW/P

On the right the code can be seen in action. The red square is the minimum area square. The text in green is the distance of the object from the camera. The accuracy in the distance is ±5 cm.

So D = 947.3x4.6/W
W = (lx+ly)/2
Where lx and ly are the sides of the red rectangle seen in the above figure

Laser offset:
Now we know D and F, and W for the laser to be 2.8 cms in the X and Y directions, we calculate P = 947.3x2.8/D

This is calculated for both x and y offsets and the laser is now instructed to move to the altered locations. The y offset was increased by an experimentally calculated value of 108 to compensate for the dip of the dart as it moves.

Once the laser reaches in a 10x10 pixel square around the altered location, the third servo is rotated by 90 degrees to trigger the gun.

Results and Precision

Currently our assembly has an accuracy of 65-70% where it hits the target head on.
The rest of the time it has a 85% accuracy, missing the edge of the target by only a couple of millimetres. Overall, the shot never goes beyond a radius of 3 cm around the centre of the target. The success rates have been tested for a range of distances from 80cm upto 150 cm.

The device is somewhat sensitive to lighting conditions. The background wall must not be very reflective and the lighting around must be moderate, otherwise the bright spots of reflection on the wall are mistaken for the laser pointer. This happens because the camera views the laser pointer as a bright white spot and bright spots of light appear the same.

In the current project, the object used as target must largely be of the same colour so that a clear contour of the object is formed with that colour. But the object could be of any colour.

Progress of the Project

Before June 1: Worked on OpenCV, and developed some basic codes to find the contours of an object of a particular colour and determine the x and y coordinates of its geometric centre in the image. The outline of the code is given above.

June 1: Review Meet 1: We worked on deciding the target to be accomplished by the second review meet, i.e. have a working pan-tilt platform with a camera and laser pointer mounted on it. The system should be such that we can click the object of a particular colour and it should point the laser at that object.

June 2,3: Went to Lamington Road to buy a Webcam, Servo motors (one 15kg-cm and one 9kg-cm), some L-clamps and screws, etc. Also bought a sheet of wood for assembling the mount.

June 3,4: Started working on creating the mount from the wood. Also, wrote the code for communication between the laptop (which will be handling the image from the camera and its processing) and the Arduino (which will rotate the servo motors for pan/tilt). This was done using the PySerial library and a Python code on the laptop to send characters to Arduino which will read it on its serial port.

June 5-8: Finished making the mount as described above. Documented most of the stuff. Tried out two versions of the program - the first one where the camera moves and brings the object to its centre; the second where only the laser moves and points at the target.

June 9-15: Finished making the final mechanical setup of the project including fixing of the gun and laser and strengthening of the existing mount using Double Sided Tape. Also purchased another Servo Motor (3kg-cm Torque) for the trigger.

June 15-20: Calibration of the Laser-Gun Combination and minor changes to the code to make the program smoother and more efficient.

Important Links:

http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_tutorials.html
Link contains all the tutorials related to Image Processing using OpenCV on Python

https://sourceforge.net/projects/numpy/files/NumPy/1.9.0/numpy-1.9.0-win32-superpack-python2.7.exe/download
Link is required for installing version 9 of NumPy as latest version of OpenCV(version 3.1) doesn't work with previous versions

http://opencv-srf.blogspot.in/2011/11/mouse-events.html
Link covers the use of setMouseCallback function to detect mouse clicks on a window
Just use cv2.EVENT_LBUTTONDOWN and so on.

http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_gui/py_mouse_handling/py_mouse_handling.html#mouse-handling
More examples on setMouseCallback

http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_colorspaces/py_colorspaces.html#converting-colorspaces
Object tracking of a particular colour of known HSV values

http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_filtering/py_filtering.html#filtering
Filtering techniques

http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_contours/py_contours_begin/py_contours_begin.html#contours-getting-started
Drawing Contours in your image

http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_contours/py_contour_features/py_contour_features.html#contour-features
For finding contour features such as centroid, area, perimeter etc.

http://www.instructables.com/id/Interface-Python-and-Arduino-with-pySerial/?ALLSTEPS
Interfacing Python with Arduino

http://www.instructables.com/id/Arduino-and-Python/step2/Python-Code/

Detecting a Laser Pointer on OpenCV:
https://bradmontgomery.net/blog/tracking-a-laser-pointer-with-python-and-opencv/

http://www.pyimagesearch.com/2015/01/19/find-distance-camera-objectmarker-using-python-opencv/
Finding depth of an object with known dimensions with a camera

https://www.youtube.com/watch?v=aswWXFbcedM
Video to convert a laser pointer's momentary switch to a toggle switch