Pan Tilt Design report

Vineet Menon, Rishabh Singh, Dr. Mangal Kothari

Abstract

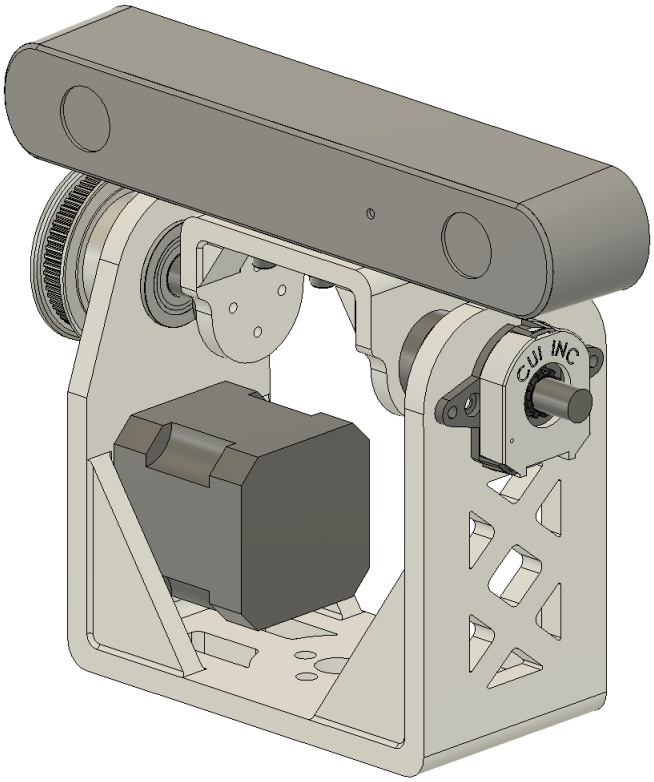
The aim of this project is to develop a 2 degree of freedom purely rotational pan-tilt mechanism which can provide stable rotation about the Yaw(for Pan) and Pitch(for Tilt) axes. This mechanism can be used for application such as terrain mapping (using a limited FOV stereo/ RGB-D camera), object tracking and as a gimbal (for stable camera feed).

1 Introduction

In this design report, the following parts are going to be covered in detail:

1. Design and fabrication
2. Component description and BOM
3. Mathematical model of system (state variable matrix, transformation matrix)
4. HSM control equations with driver
5. System architecture

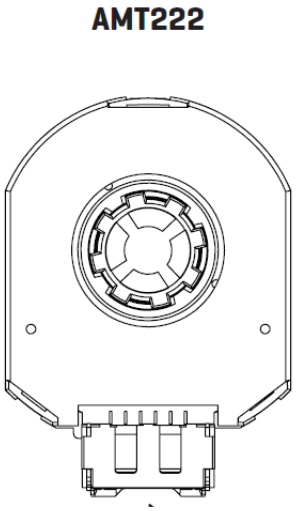
2 Design and Fabrication



3 Components used in the Tilt mechanism-

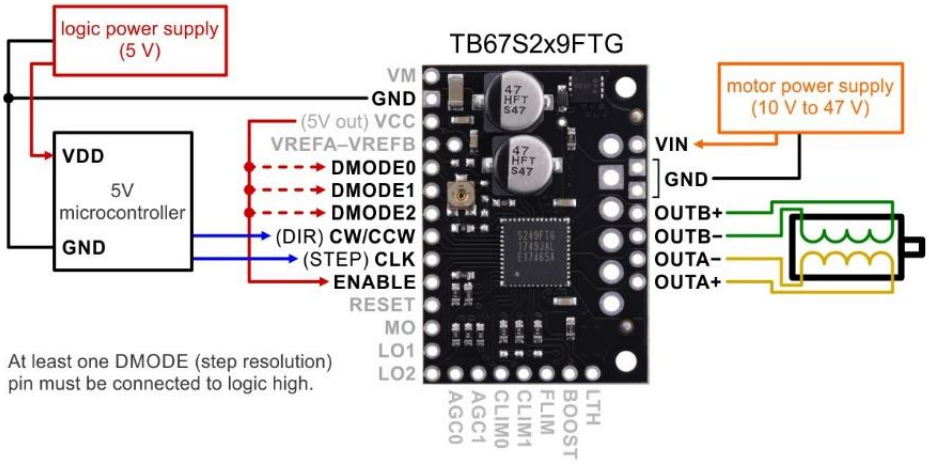
1. Stepper motor
    - Step angle = 1.8 degree/pulse or 200 pulse/revolution
    - Rated voltage = 12V
    - Rated current = 0.4A
    - Phase resistance = 30Ω
    - Phase Inductance = 60mH
    - Holding torque = 4.2kg-cm
    - Detent torque = 220g.cm
    - Rotor inertia = $54g.cm^2$
    - Weight = 0.28kg
2. CUI AMT22 MODULAR ABSOLUTE ENCODER
    - Full duplex Serial Peripheral Interface (SPI)
    - 12-bit absolute position
    - Checksum bits for error detection
    - Digitally settable zero position

PINOUT CONNECTOR	
#	Function
1	+5 V
2	SCLK
3	MOSI
4	GND
5	MISO
6	CHIP SELECT



3. IMU (accelerometer and gyroscope) -
   6-axis Gyroscope and Accelerometer used to acquire the angular velocity and linear acceleration of the camera frame w.r.t. the ground frame.

4. Stepper motor driver

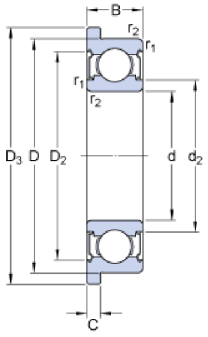


5. Timing Pulley and Belt

   - Driver pulley properties-
     - No. of teeth: 15
     - Pitch diameter: 9.70mm
   - Driven pulley properties-
     - No. of teeth: 60
     - Pitch diameter: 38.81mm
   - Timing belt properties-
     - No of grooves: 90
     - Centre-to-centre distance: 51.4 mm

6. Flanged ball bearings -

Dimensions



d		8	mm
D		22	mm
B		7	mm
d_2	≈	10.5	mm
D_2	≈	19.03	mm
D_3		25	mm
C		1.5	mm
$r_{1,2}$	min.	0.3	mm

4 Mathematical model of system

The nonlinear model based on the Lagrange-Euler equation is as follows.

$$\mu = M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + F(\dot{\theta}) + g(\alpha)$$

Here $\mu = \tau$ is control input and θ is feedback input vector representing the joint orientation, $M(\theta)$ is an inertia matrix and $C(\theta, \dot{\theta})$ accounts for centrifugal and coriolis forces. The term $F(\dot{\theta})$ accounts for the viscous friction. The term $g(\alpha)$ accounts for gravity forces, here α is angle between the arm length and force (mg). The term $C(\theta, \dot{\theta})$ is effective when the mass centre begins to move away from the centre of rotation. The effect of centrifugal force becomes evident on the distributed mass of the body. Therefore, centre of PTP is located on the rotational axis of the body and term $C(\theta, \dot{\theta})$ is neglected in linear model. Thus the reduced non-linear-system equation for single mechanism (joint) is given as:

$$\tau = J_{eff}\ddot{\theta} + f_{eff}\dot{\theta} + mglsin\theta$$

Here, τ = Torque, J_{eff} = Effective Inertia Load, f_{eff} = viscous friction coefficient

This is a general equation for both the joints and represents the non-linear model of pan and tilt mechanism. The viscous friction is a frictional force that resists objects in motion. The viscous friction is actually a property of the medium in which the motion of the object is occurring. Any fluid medium, such as the grease in the bearings or the air, has an internal resistance to flow, which is represented by the viscous friction. The effective inertial loads and viscous friction coefficient of the system are computed using following relation [1]:

$$J_{eff} = J_m + n^2 J_l, f_{eff} = f_m + n^2 f_l$$

Here, J_m = Motor Inertia, J_l = Load Inertia, n = Gear ratio between motor and load

5 Hybrid Stepper Motor Control equations -

Stepper motors are controlled using off-the shelf driver modules which acquires a square pulse as input from a microcontroller or a timer circuit and based on time period of the single pulse, the motor rotates by a certain step (1.8 degree) or microstep (factors of 1.8). A trail of pulse is required to rotate the motor to a desired angle.

$$\alpha = F[M, Tout]$$

Where,
α = required step/micro-step angle
M= desired microstepping factor
Tout = Time period of the input square pulse

5.1 Micro-stepping angle-

The default step angle of our stepper motor is 1.8 degrees per pulse. This indicates that when microstepping factor(M) is unity, the shaft of the motor will rotate by 1.8 degree when a square pulse is applied at the 'STEP' pin of the driver.

In certain drivers, there is an feature called "Micro-stepping" using which we can reduce this step angle to finer values like 0.9 degrees, 0.45 degrees and so on. The general equation for micro-stepping can be written as -

$$\alpha = 1.8/M$$

This equation will yield the microstep angle α in degrees. To maintain our calculations in S.I. units, we convert this value to radians using the following conversion -

$$\alpha = (1.8/M) * (\pi/180)$$

$$\alpha = \pi/(100M)$$

A point to be noted here is that since we are using a pulley-belt reduction mechanism, our load(camera) will be rotating 4 times slower than the motor. This means that if the motor's step angle changes by α, the load's angle would change by $\alpha^{'}$ and $\alpha^{'}$ can be denoted as -

$$\boxed{\alpha^{'} = \pi/(400M)} \quad (1)$$

5.2 Angular Speed of the motor -

For simplicity, we will enter the maximum desired angular rotation ω of the load in terms of rpm (revolutions per minute). But in order to keep all calculations in S.I. unit, we calculate $\omega^{'}$ which represents angular rotation in rad/sec -

$$\boxed{\omega^{'} = [(2\pi)/60] * \omega} \tag{2}$$

5.3 Frequency of the square pulse to be generated -

Since we now have the values for $\alpha^{'}$ and $\omega^{'}$, we now calculate the Frequency (Fout) and Time period (Tout) of the square pulse needed to be generated.

$$\omega^{'} = \alpha^{'}/Tout$$
$$\omega^{'} = \alpha^{'} * Fout$$
$$Fout = \omega^{'}/\alpha^{'}$$
$$Fout = [(2\pi/60) * \omega] * [(400M)/\pi]$$
$$\boxed{Fout = (40/3) * M\omega} \tag{3}$$

5.4 Determine the number of pulses -

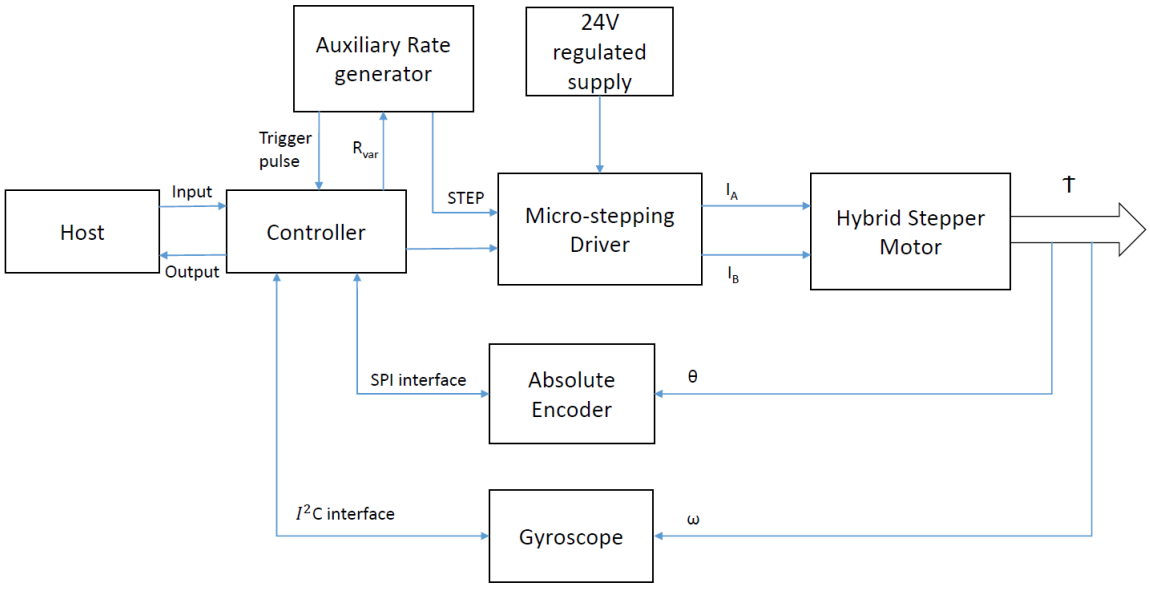
Now with the calculated value of Fout or Tout, we know that on applying one such pulse the motor will rotate by a step or micro-step. But if we want the motor to rotate to a certain angle say θ_{final}, we need to send a trail of square pulses which would continuously rotate the shaft of the motor in discrete micro-steps at a speed depending on the value of Tout.
To determine this value of pulse count (pls_{count}), we make a simple comparison as follows -
If one pulse of frequency Fout corresponds to $\alpha^{'}$ where $\alpha^{'} = \pi/(400M)$;
'pls_{count}' number of pulses of frequency Fout should correspond to θ_{final}
On comparison we get,

$$pls_{count} * \alpha^{'} = \theta_{final}$$

$$pls_{count} = \frac{(4M) * \theta_{final}}{1.8}$$

$$\boxed{pls_{count} = \frac{20 * M\theta_{final}}{9}} \tag{4}$$

6 System Implementation-



The goal of our system is to ensure that the load is rotated to the required angle while never exceeding the specified rotation speed limit (in rpm). Both the threshold angles and maximum speed are specified by the user. The following are the electronic components use to implement and ensure that the stepper motor is properly controlled as per the angular velocity and angle commanded by the PID controller.

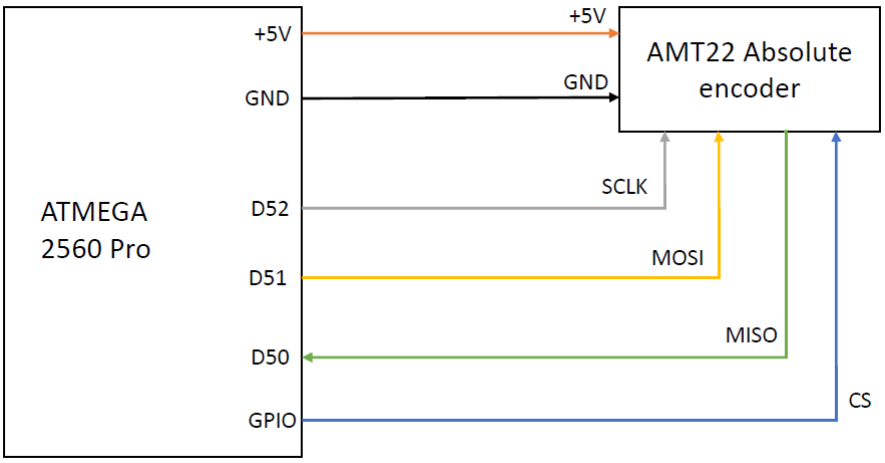
- Host-
  - Function: To acquire input parameters from the user (w(max), M, endpoint angles [Theta(max), Theta(min)]), calculate the values of Fout and pls_{count} and pass on the calculated values to the controller.
- Controller-
  - Function: The controller acts as a bridge between the host and the subordinate components like feedback sensors, stepper motor controller and auxiliary rate generator circuit. We will be using Atmega 2560 Pro for basic prototyping, but eventually replace it with a STM32F4 development board.
- Sensors-
  - Absolute encoder: Connected to the output shaft of the load. The output is read by the micro-controller in the form of a 12-bit binary value which represents the absolute angular position of the load driven by the effect of the driven pulley. This is indeed the source of angular position feedback.
  - Gyroscope: Interfaced directly with the frame connected to the load. This will provide angular velocity (in rad/sec) with respect to the Y-axis of the IMU (pitch axis). This is indeed the source of angular velocity feedback.
- Auxiliary rate generator-

– Function: The rate generator circuit will be designed such that depending on the required value of pulse frequency (Fout), a square pulse would be generated by the timer circuit which is then feed to the "STEP" pin of the stepper motor driver.

6.1 Micro-controller interfacing with peripheral components-

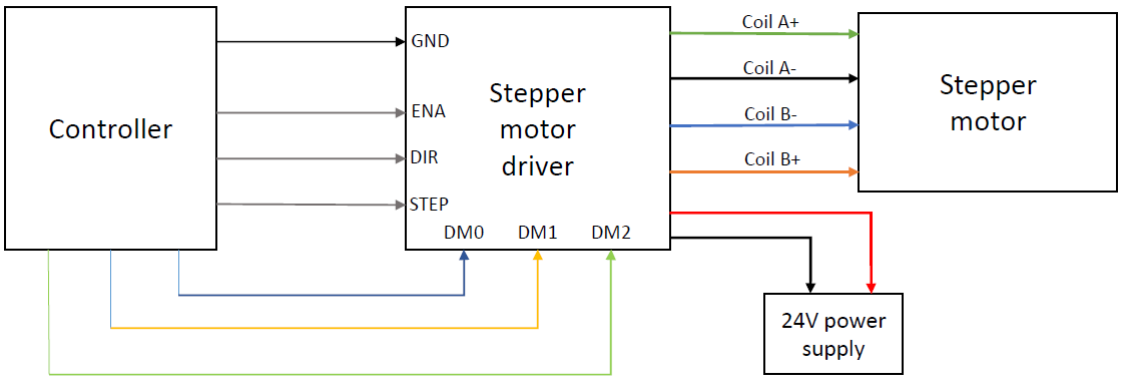
6.1.1 With Absolute encoder -

The transmission of data from absolute encoder is accomplished in the form of a synchronous serial data protocol known as SPI(Serial Peripheral Interface) as described in [6]. The connections from the encoder to the Arduino based micro-controller are made with reference to guidelines provided in [3]. It is as shown in the figure below-



The absolute encoder datasheet and the sample reference program provided by CUI [4] were thoroughly considered to properly interface the encoder with the micro-controller.

6.1.2 With Stepper motor and driver-



The stepper motor driver which we are using is a Toshiba TB67S249FTG based compact carrier board. This carrier board enables us to regulate the speed and alter the direction of the stepper motor through command signals provided by the micro-controller to the "STEP" and "DIR" pins of the driver respectively. The program logic for the same was executed on the same Atmega 2560 controller which is used to read the encoder data. The guidelines provided by the

manufacturer [5] were referred thoroughly while writing the embedded program for the control of the stepper program.

6.1.3 Timer generation function -

In [2], a PD frequency control is adopted as means of setting prescaler value of STM32 timer.

References

[1] Madan Gopal. *Control systems: principles and design*. Tata McGraw-Hill Education, 2002.

[2] R. Zhang et al. "Control system design for two-wheel self-balanced robot based on the stepper motor". In: *Proceedings of 2013 IEEE International Conference on Service Operations and Logistics, and Informatics*. Dongguan, China, 2013, pp. 241–244. DOI: 10.1109/SOLI.2013.6611417.

[3] Arduino. *SPI library*. URL: https://www.arduino.cc/en/reference/SPI. (accessed: 08.03.2020).

[4] CUI Devices. *Absolute Encoder*. URL: https://www.cuidevices.com/product/motion/rotary-encoders/absolute/modular/amt22-series. (accessed: 08.03.2020).

[5] Pololu. *TB67S249FTG Stepper Motor Driver*. URL: https://www.pololu.com/product/3096. (accessed: 08.03.2020).

[6] Sparkfun. *Serial Peripheral Interface (SPI)*. URL: https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi/all. (accessed: 08.03.2020).