AIM

As the name suggests, the robot would be a bicycle balancing autonomously.The self stabilizing bicycle will employ a control system to keep itself from falling over while in motion.

INTRODUCTION

Bicycles are a common form of exercise, recreation and transportation used by billions. They can also serve to provide physical therapy, as they are a low impact form of exercise that can train balance, strength, stamina and coordination. Though one may consider riding a bicycle to be a fairly simple task, this is not the case for many people. This includes young childeren, adults who have never learned to ride a bicycle, injured people, or people sufferring from developmental or cognitive disabilities. A system that could provide balancing assistance to a bicycle rider without otherwise affecting the experience of riding a bicycle could provide great benefit to these groups of individuals. Such a system could be used both as a teaching tool, and as a physically theraputic device.

This problem of balancing a bicycle is analogous to what is known as the 'inverted pendulum' problem. An inverted pendulum is a pendulum which has its mass above its pivot (figure 1). The pendulum can be anything from a simple mass and rod, to a full system. While a normal pendulum is stable, an inverted pendulum is inherently unstable, and must be actively balanced to remain upright. In the case of a bicycle, the bicycle is a rigid body which can rotate around its contact point with the ground. Although a bicycle motion has multiple degrees of freedom, the particular type of motion which this project aimsto stabilize is this tilt angle around the point of contact with the ground relative to the direction of gravity

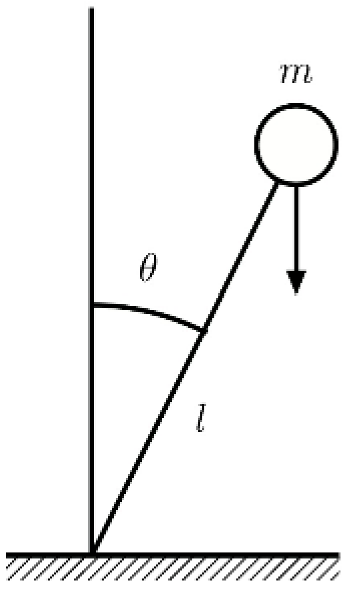


Figure 1: Inverted Pendulum

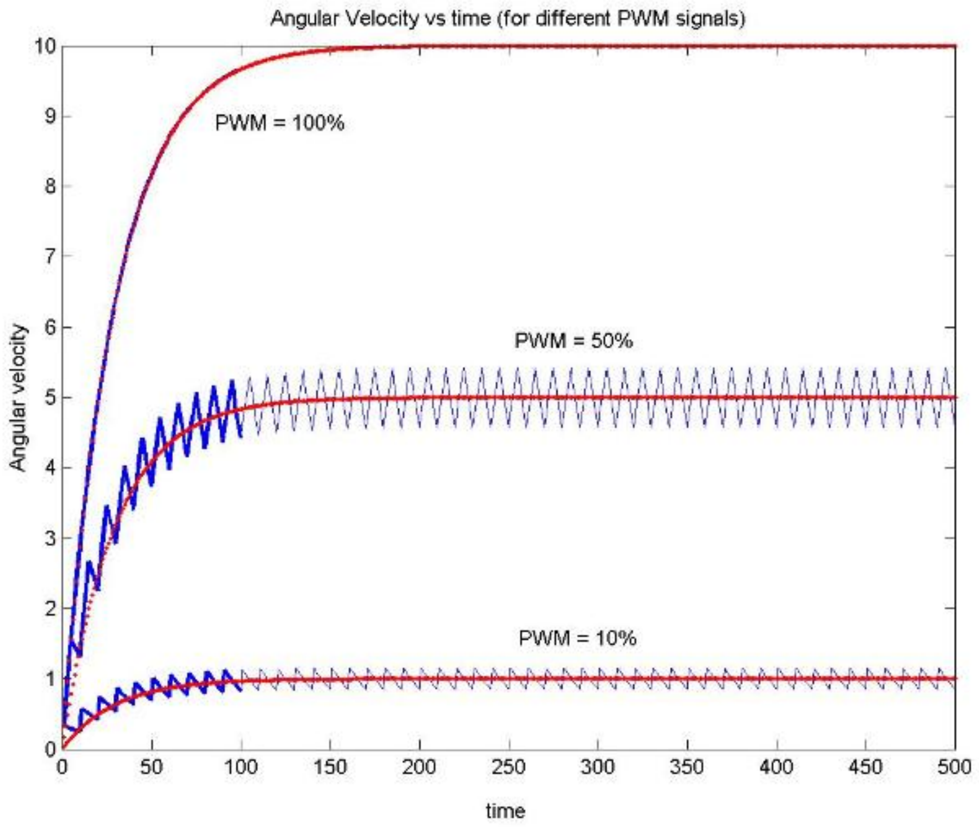
Why not just make a 4-wheeled or 3-wheeled robot?
The following aresome reasons:

•A bicycle uses less track space as compared to any other vehicle.
•It is more flexible and easy to steer as compared to any other vehicle.
•It can cut through obstacles easier than a four wheeled bot.
•This technology can be extended to make a self balancing unicycle which is much better in terms of space and steer ability.
•Due to its low weight and greater steer ability, it can be made faster than a humanoid robot.
•This robot might be useful for transportation in research areas or mines where man or four wheeled robots cannot reach.

Principle-

The basic principle is that when we apply torque on the flywheel, the cycle experiences a reactionary torque. So, increasing the RPM in the direction of falling cycle produces a reactionary torque on the bicycle to lift it up.

Relationship between PWM, RPM and time-



$$\omega = \omega_f(1 - e^{-\frac{T_s}{J\omega_f}t})$$

$$\omega_f = \frac{k_t V}{k_e R}$$

$$T_s = \frac{k_t V}{R}$$

What we can observe from the graph is that the torque experienced will be maximum in the beginning (when the RPM is less).

In our case, to calculate the PWM at each instant using mathematics-

$$I*\alpha = K\ m\ g\ l\ \mathrm{Sin}\Theta$$

I= Moment of inertia of flywheel

α= Angular acceleration of the bicycle

K= constant

m= mass of the bicycle

g= 9.8 m/s^2

Θ= Angle of inclination of bicycle

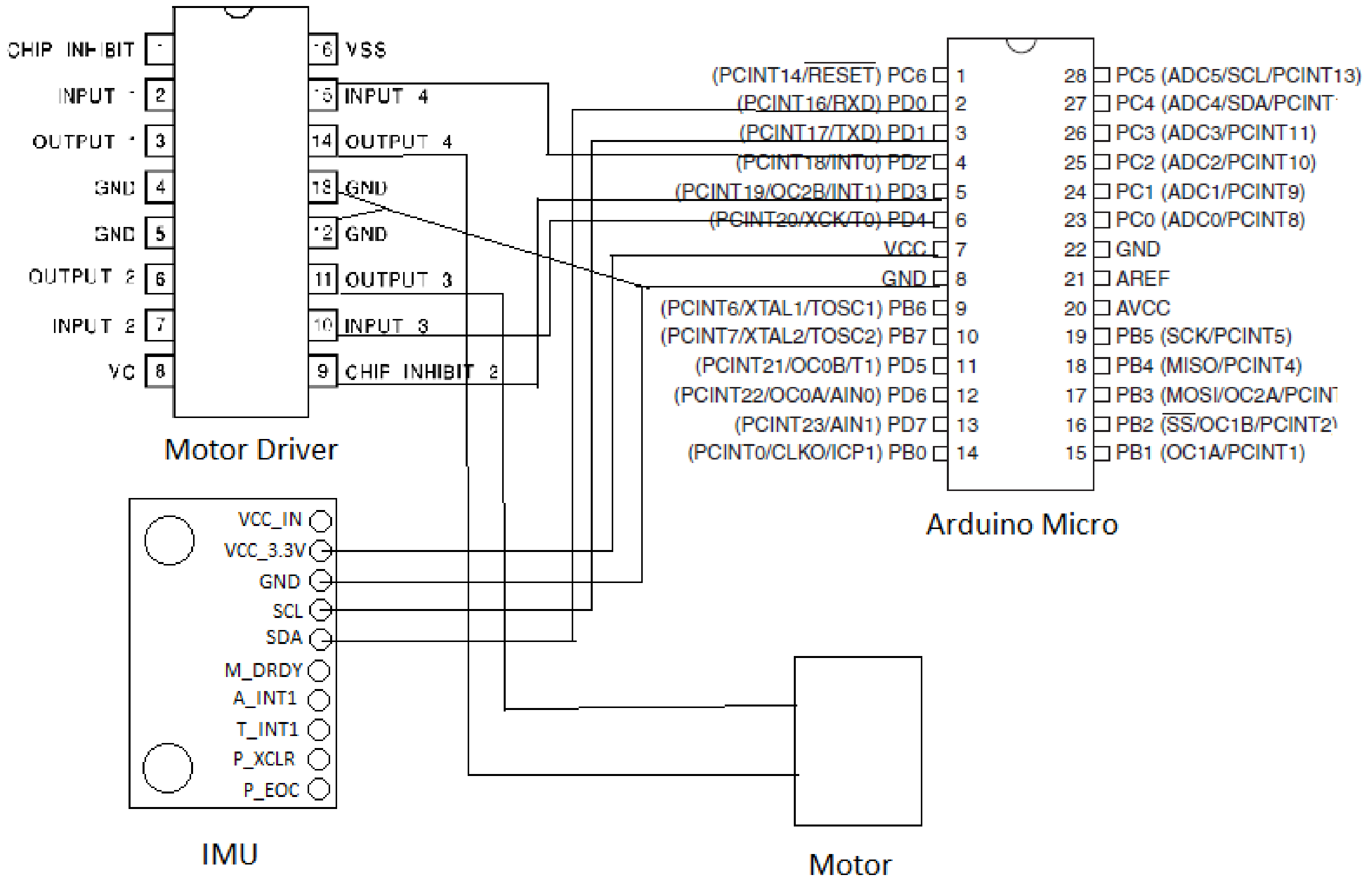
Using this equation we would have to solve a differential equation at each instant to find the correct PWM required, which is quite complicated. So, we can use a PID Controller instead.

The PWM supplied is proportional to the angle of inclination of the bicyle

COMPONENTS

- **Sensor** –GY-80 IMU- 10 DOF with L3G4200D (3-Axis Angular Rate Sensor), ADXL345 (3-Axis Digital Accelerometer), HMC5883L (3-Axis Digital Compass) and BMP085 (Barometric Pressure Sensor)
- **Micro Controller**- Arduino Micro
- **Motors**- Low torque motor for driving (60 RPM 12V DC Motor), High torque motor for flywheel(specifications)
- **Motor Driver-** L298
- Chassis and Reaction Wheel - Aluminium
- Wheels(4 cm)

CONNECTIONS



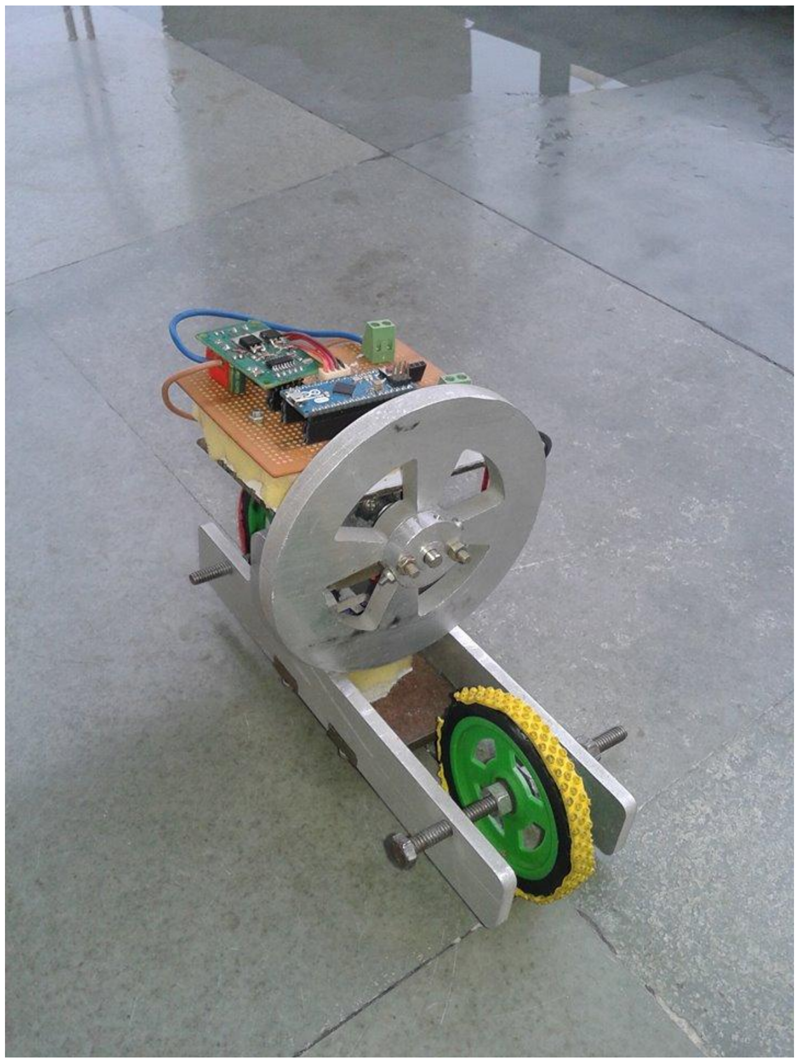
MECHANICAL DESIGN

The skeleton of the cycle is made of Aluminium, with dead Iron attached in the bottom and the motor is mounted on Nylon.

Circuit is mounted above the motor on Mica Sheet.

The design is such that :

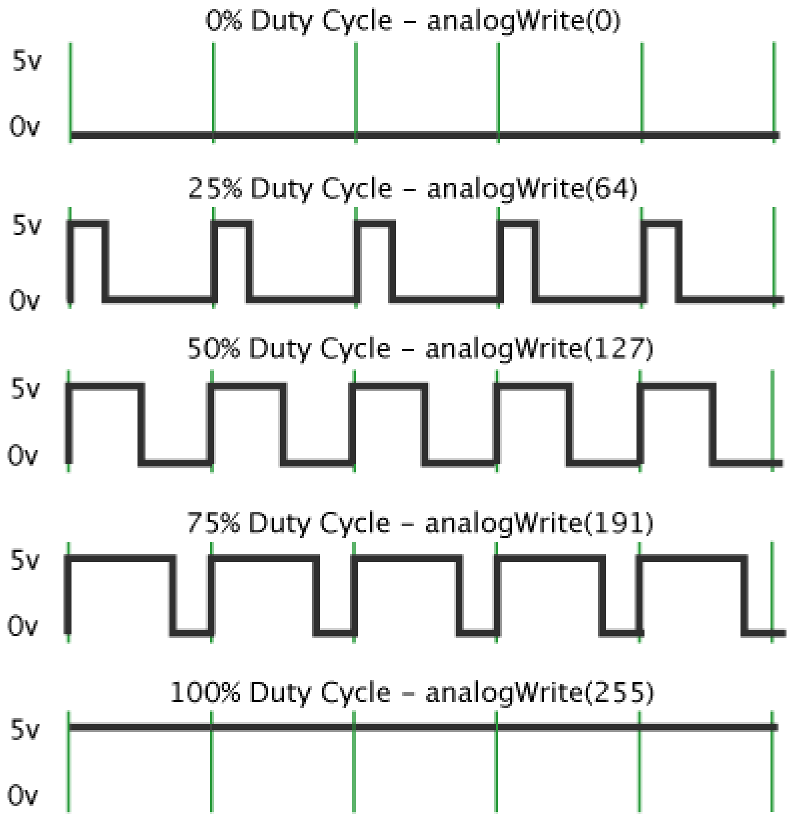
1. It has lower Centre of Mass.
2. Low induced vibrations due to motor.
3. Wheels are aligned with Shaft’s axis.
4. Is symmetrical about vertical plane.
5. The axis of Reaction Wheel is above centre of mass of whole system.



MOTORS

Motor Driver and H-BRIDGE-

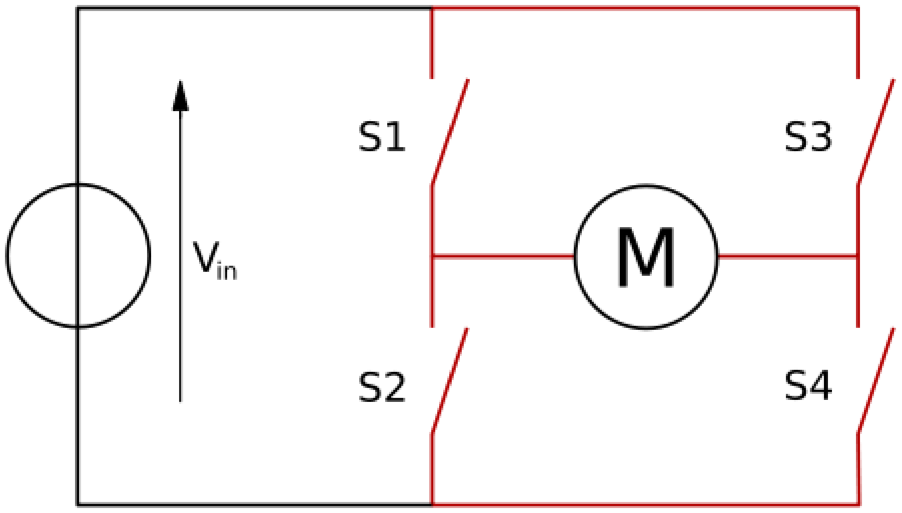
Motor Driver in combination with arduino is used to control the voltage supply of the battery to the motor. It maps the 0 to 5 V scale of arduino to the voltage supply of the battery and thus Pulse Width Modulation(PWM) can alter the average voltage supply to the motor



$$Duty\ Cycle = \frac{T_{ON}}{T_{ON} + T_{OFF}} = \frac{T_{ON}}{Time\ Period}$$

The H-bridge arrangement in motor driver is generally used to reverse the polarity of the motor, but can also be used to 'brake' the motor, where the motor comes to a sudden stop, as the motor's terminals are shorted, or to let the motor 'free run' to a stop, as the motor is effectively disconnected from the circuit. The following table summarises operation, with S1-S4 corresponding to the diagram.

S1	S2	S3	S4	Result
1	0	0	1	Motor moves right
0	1	1	0	Motor moves left
0	0	0	0	Motor free runs
0	1	0	1	Motor brakes
1	0	1	0	Motor brakes



S1,S2,S3 and S4 are actually transistors which have two states – “on” and “off”.Supplying high voltage to the transistor closes the corresponding switch.

TAKING READINGS(IMU)

Calibrating accelerometer –

We have to calibrate the accelerometerwith the help of force of gravity. This means that we have to find the sensitivity of accelerometer and the bias of the accelerometer in each axis.

For X Axis-

$\text{Acceleration}_x = (\text{Sensor}_x - \text{Bias}_x)/\text{Senstivity}_x$

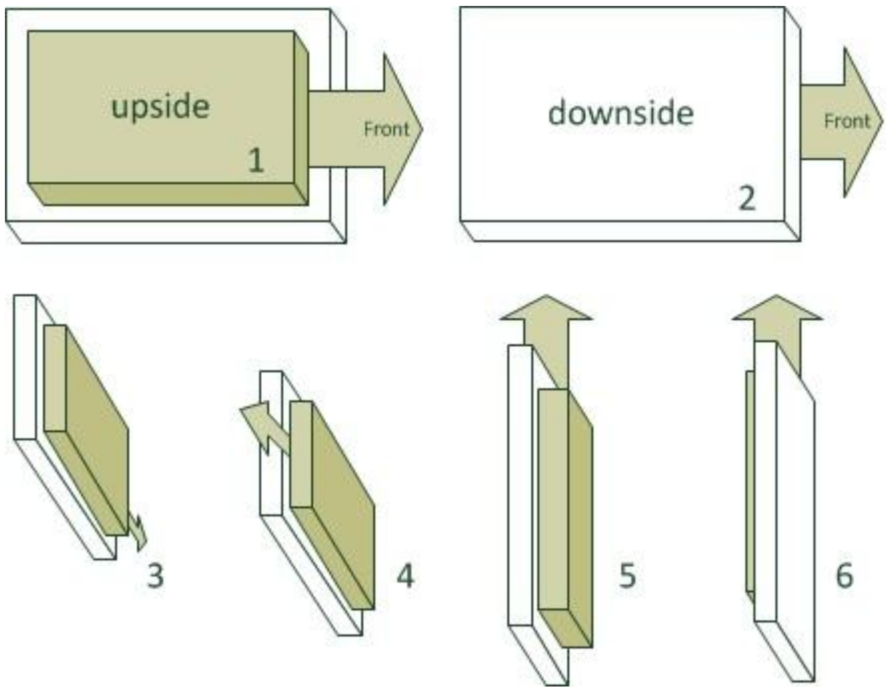
STEPS-

1. Hold the IMU in such a way that the Z-axis of the IMU is perpendicular to the ground. A cuboid can be used for this purpose.
2. Record the reading from the sensor along z direction,say $Z_{max.}$
3. Invert the IMU and record the reading from the sensor along z direction,say $Z_{min.}$
4. Along Z direction-

   $\text{Senstivity}_z = (Z_{max} - Z_{min})/2$

   $\text{Bias}_z = (Z_{max} - Z_{min})/2$

5. Repeat the above steps by keeping IMU's X and Y axis perpendicular to the ground.



Calibrating Gyroscope-

Gyroscope gives the value of angular velocity along the 3 axis. So we only have to find the value of bias in all the 3 directions for calibrating the gyroscope.

Along X axis-

$\omega_x = \text{Sensor}_x - \text{Bias}_x$

STEPS-

1. Hold the IMU still so that there is no other force experienced by the IMU except gravitational force
2. Record the readings of the sensors along the three axis. These are the bias anglular velocities

Filtering of raw data from accelerometer and gyroscope-

A low pass filter is used to remove the high frequency disturbances in the readings of accelerometer. This type of filter attenuates the higher frequencies of the signal, thus providing a smoother reading. The Low-Pass filter is easily implemented by using the following equation:

$$y_t = \alpha * x_t + (1-\alpha) * y_{t-1}$$

y_t = filtered reading

y_{t-1}= previously filtered reading

x_t = accelerometer reading

α = smoothing factor

Here α should be not much less than 1 otherwise there will be a drift

From the actual reading over time

Calculation of angles-

The accelerometer gives us data relative to the global coordinate system while gyroscope gives data relative to the position of the IMU. Direction Cosine Matrix(DCM) is used to convert the vector from one coordinate system to another just by multiplying by a matrix.

Part 1. The DCM Matrix

Generally speaking orientation kinematics deals with calculating the relative orientation of a body relative to a global coordinate system. It is useful to attach a coordinate system to our body frame and call it Oxyz, and another one to our global frame and call it OXYZ. Both the global and the body frames have the same fixed origin O (see *Fig. 1*). Let's also define **i,j, k** to be unity vectors co-directional with the body frame's x, y, and z axes- in other wordsthey are versors of Oxyz and let **I, J, K** be the versors of global frame OXYZ.

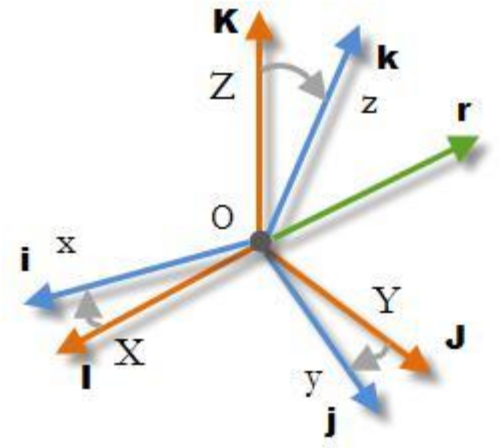


Figure 1

Thus, by definition, expressed **in terms of global coordinates** vectors **I, J, K** can be written as:

$\mathbf{I}^G = \{1,0,0\}^{\mathbf{T}}$, $\mathbf{J}^G=\{0,1,0\}^{\mathbf{T}}$, $\mathbf{K}^G = \{0,0,1\}^{\mathbf{T}}$

Note: we use $\{\ldots\}^T$ notation to denote a column vector, in other words a column vector is a translated row vector. The orientation of vectors (row/column) will become relevant once we start multiplying them by a matrix later on in this text.

And similarly, **in terms of body coordinates** vectors **i, j, k** can be written as:

$\mathbf{i}^B = \{1,0,0\}^{\mathbf{T}}$, $\mathbf{j}^B=\{0,1,0\}^{\mathbf{T}}$, $\mathbf{k}^B = \{0,0,1\}^{\mathbf{T}}$

Now let's see if we can write vectors **i, j, k** in terms of global coordinates. Let's take vector **i** as an example and write its global coordinates:

$\mathbf{i}^G = \{i_x{}^G, i_y{}^G, i_z{}^G\}^{\mathbf{T}}$

Again, by example let's analyze the X coordinate $i_x{}^G$, it's calculated as the length of projection of the **i** vector onto the global X axis.

$i_x{}^G = |\mathbf{i}| \cos(X,\mathbf{i}) = \cos(\mathbf{I},\mathbf{i})$

Where $|\mathbf{i}|$ is the norm (length) of the **i** unity vector and $\cos(\mathbf{I},\mathbf{i})$ is the cosine of the angle formed by the vectors **I** and **i**. Using the fact that $|\mathbf{I}| = 1$ and $|\mathbf{i}| = 1$ (they are unit vectors by definition). We can write:

$i_x^G = \cos(\mathbf{I},\mathbf{i}) = |\mathbf{I}||\mathbf{i}| \cos(\mathbf{I},\mathbf{i}) = \mathbf{I}.\mathbf{i}$

Where **I.i**. is the scalar (dot) product of vectors **I** and **i**. For the purpose of calculating scalar product **I.i** it doesn't matter in which coordinate system these vectors are measured as long as they are both expressed in the same system, since a rotation does not modify the angle between vectors so: $\mathbf{I}.\mathbf{i} = \mathbf{I}^B.\mathbf{i}^B = \mathbf{I}^G.\mathbf{i}^G = \cos(\mathbf{I}^B.\mathbf{i}^B) = \cos(\mathbf{I}^G.\mathbf{i}^G)$, so for simplicity we'll skip the superscript in scalar products **I.i , J.j , K.k** and in cosines cos(**I**,**i**), cos(**J**,**j**), cos(**K**,**k**).

Similarly we can show that:

$i_y^G = \mathbf{J}.\mathbf{i}$, $i_z^G = \mathbf{K}.\mathbf{i}$, so now we can write vector **i** in terms of global coordinate system as:

$\mathbf{i}^G = \{ \mathbf{I}.\mathbf{i}, \mathbf{J}.\mathbf{i}, \mathbf{K}.\mathbf{i}\}^T$

Furthermore, similarly it can be shown that $\mathbf{j}^G = \{ \mathbf{I}.\mathbf{j}, \mathbf{J}.\mathbf{j}, \mathbf{K}.\mathbf{j}\}^T$, $\mathbf{k}^G = \{ \mathbf{I}.\mathbf{k}, \mathbf{J}.\mathbf{k}, \mathbf{K}.\mathbf{k}\}^T$.

We now have a complete set of global coordinates for our body's versors **i, j, k** and we can organize these values in a convenient matrix form:

$$[\mathbf{i}^G, \mathbf{j}^G, \mathbf{k}^G] = \begin{bmatrix} \mathbf{I}.\mathbf{i} & \mathbf{I}.\mathbf{j} & \mathbf{I}.\mathbf{k} \\ \mathbf{J}.\mathbf{i} & \mathbf{J}.\mathbf{j} & \mathbf{J}.\mathbf{k} \\ \mathbf{K}.\mathbf{i} & \mathbf{K}.\mathbf{j} & \mathbf{K}.\mathbf{k} \end{bmatrix} = \begin{bmatrix} \cos(\mathbf{I},\mathbf{i}) & \cos(\mathbf{I},\mathbf{j}) & \cos(\mathbf{I},\mathbf{k}) \\ \cos(\mathbf{J},\mathbf{i}) & \cos(\mathbf{J},\mathbf{j}) & \cos(\mathbf{J},\mathbf{k}) \\ \cos(\mathbf{K},\mathbf{i}), & \cos(\mathbf{K},\mathbf{j}) & \cos(\mathbf{K},\mathbf{k}) \end{bmatrix} = \mathrm{DCM}^G$$

(Eq. 1.1)

This matrix is called Direction Cosine Matrix for now obvious reasons - it consists of cosines of angles of all possible combinations of body and global versors.

The task of expressing the global frame versors $\mathbf{I}^G$, $\mathbf{J}^G$, $\mathbf{K}^G$ in body frame coordinates is symmetrical in nature and can be achieved by simply swapping the notations **I, J, K** with **i, j,k,** the results being:

$\mathbf{I}^B = \{ \mathbf{I}.\mathbf{i}, \mathbf{I}.\mathbf{j}, \mathbf{I}.\mathbf{k}\}^T$, $\mathbf{J}^B = \{ \mathbf{J}.\mathbf{i}, \mathbf{J}.\mathbf{j}, \mathbf{J}.\mathbf{k}\}^T$, $\mathbf{K}^B = \{ \mathbf{K}.\mathbf{i}, \mathbf{K}.\mathbf{j}, \mathbf{K}.\mathbf{k}\}^T$

and organized in a matrix form:

$$[\mathbf{I}^B, \mathbf{J}^B, \mathbf{K}^B] = \begin{bmatrix} \mathbf{I}.\mathbf{i} & \mathbf{J}.\mathbf{i} & \mathbf{K}.\mathbf{i} \\ \mathbf{I}.\mathbf{j} & \mathbf{J}.\mathbf{j} & \mathbf{K}.\mathbf{j} \\ \mathbf{I}.\mathbf{k} & \mathbf{J}.\mathbf{k} & \mathbf{K}.\mathbf{k} \end{bmatrix} = \begin{bmatrix} \cos(\mathbf{I},\mathbf{i}) & \cos(\mathbf{J},\mathbf{i}) & \cos(\mathbf{K},\mathbf{i}) \\ \cos(\mathbf{I},\mathbf{j}) & \cos(\mathbf{J},\mathbf{j}) & \cos(\mathbf{K},\mathbf{j}) \\ \cos(\mathbf{I},\mathbf{k}), & \cos(\mathbf{J},\mathbf{k}) & \cos(\mathbf{K},\mathbf{k}) \end{bmatrix} = \mathrm{DCM}^{\mathrm{B}}$$

(Eq. 1.2)

It is now easy to notice that $\mathrm{DCM}^B = (\mathrm{DCM}^G)^T$ or $\mathrm{DCM}^G = (\mathrm{DCM}^B)^T$, in other words the two matrices are translates of each other, we'll use this important property later on.

Also notice that $\mathrm{DCM}^B. \mathrm{DCM}^G = (\mathrm{DCM}^G)^T .\mathrm{DCM}^G = \mathrm{DCM}^B. (\mathrm{DCM}^B)^T = I_3$, where I_3 is the 3x3 identity matrix. In other words the DCM matrices are orthogonal.

This can be proven by simply expanding the matrix multiplication in block matrix form:

$$\mathrm{DCM}^{\mathrm{B}}\mathrm{DCM}^{\mathrm{G}} = \mathrm{DCM}^{\mathrm{GT}}\mathrm{DCM}^{\mathrm{G}} = \begin{bmatrix} \mathbf{i}^{GT} \\ \mathbf{j}^{GT} \\ \mathbf{k}^{GT} \end{bmatrix} [\mathbf{i}^G, \mathbf{j}^G, \mathbf{k}^G] = \begin{bmatrix} \mathbf{i}^{GT}.\mathbf{i}^G & \mathbf{i}^{GT}.\mathbf{j}^G & \mathbf{i}^{GT}.\mathbf{k}^G \\ \mathbf{j}^{GT}.\mathbf{i}^G & \mathbf{j}^{GT}.\mathbf{j}^G & \mathbf{j}^{GT}.\mathbf{k}^G \\ \mathbf{k}^{GT}.\mathbf{i}^G & \mathbf{k}^{GT}.\mathbf{j}^G & \mathbf{k}^{GT}.\mathbf{k}^G \end{bmatrix} =$$
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = I_3 \quad \textit{(Eq. 1.3)}$$

To prove this we use such properties as for example: $\mathbf{i}^{GT}. \mathbf{i}^G = | \mathbf{i}^G|| \mathbf{i}^G|\cos(0) = 1$ and $\mathbf{i}^{GT}. \mathbf{j}^G = 0$ because ($\mathbf{i}$ and $\mathbf{j}$ are orthogonal) and so forth.

The DCM matrix (also often called the rotation matrix) has a great importance in orientation kinematics since it defines the rotation of one frame relative to another. It can also be used to determine the global coordinates of an arbitrary vector if we know its coordinates in the body frame (and vice versa).

Let's consider such a vector with body coordinates:

$\mathbf{r}^B = \{ r_x^B, r_y^B, r_z^B\}^T$and let's try to determine its coordinates in the global frame,by using aknown rotation matrix DCM^G.

We start by doing following notation: $\mathbf{r}^G = \{ r_x^G, r_y^G, r_z^G\}^T$.

Now let's tackle the first coordinate r_x^G:

$r_x^G = | \mathbf{r}^G| \cos(\mathbf{I}^G,\mathbf{r}^G)$, because r_x^G is the projection of $\mathbf{r}^G$ onto X axis that is co-directional with $\mathbf{I}^G$.

Next let's note that by definition a rotation is such a transformation that does not change the scale of a vector and does not change the angle between two vectors that are subject to the same rotation, so if we express some vectors in a different rotated coordinate system the norm and angle between vectors will not change:

$| \mathbf{r}^G| = | \mathbf{r}^B|$, $| \mathbf{I}^G| = | \mathbf{I}^B| = 1$ and $\cos(\mathbf{I}^G,\mathbf{r}^G) = \cos(\mathbf{I}^B,\mathbf{r}^B)$, so we can use this property to write

$r_x^G = | \mathbf{r}^G| \cos(\mathbf{I}^G,\mathbf{r}^G) = | \mathbf{I}^B || \mathbf{r}^B| \cos(\mathbf{I}^B,\mathbf{r}^B) = \mathbf{I}^B . \mathbf{r}^B = \mathbf{I}^B . \{ r_x^B, r_y^B, r_z^B\}^T$, by using one the two definition of the scalar product.

Now recall that $\mathbf{I}^B = \{ \mathbf{I.i, I.j, I.k}\}^T$ and by using the other definition of scalar product:

$r_x^G = \mathbf{I}^B . \mathbf{r}^B = \{ \mathbf{I.i, I.j, I.k}\}^T . \{ r_x^B, r_y^B, r_z^B\}^T = r_x^B\mathbf{I.i} + r_y^B\mathbf{I.j} + r_z^B\mathbf{I.k}$

In same fashion it can be shown that:

$r_y^G = r_x^B\mathbf{J.i} + r_y^B\mathbf{J.j} + r_z^B\mathbf{J.k}$
$r_z^G = r_x^B\mathbf{K.i} + r_y^B\mathbf{K.j} + r_z^B\mathbf{K.k}$

Finally let's write this in a more compact matrix form:

$$\mathbf{r}^G = \begin{bmatrix} \mathbf{r}_x^G \\ \mathbf{r}_y^G \\ \mathbf{r}_z^G \end{bmatrix} = \begin{bmatrix} \mathbf{I.i} & \mathbf{I.j} & \mathbf{I.k} \\ \mathbf{J.i} & \mathbf{J.j} & \mathbf{J.k} \\ \mathbf{K.i} & \mathbf{K.j} & \mathbf{K.k} \end{bmatrix} \begin{bmatrix} \mathbf{r}_x^B \\ \mathbf{r}_y^B \\ \mathbf{r}_z^B \end{bmatrix} = \mathrm{DCM}^G \mathbf{r}^B \qquad \textit{(Eq. 1.4)}$$

Thus the DCM matrix can be used to covert an arbitrary vector $\mathbf{r}^B$ expressed in one coordinate system B, to a rotated coordinate system G.

We can use similar logic to prove the reverse process:

$$\mathbf{r}^B = \mathrm{DCM}^B \mathbf{r}^G \quad \textit{(Eq. 1.5)}$$

Or we can arrive at the same conclusion by multiplying both parts in *(Eq. 1.4)* by DCM^B which equals to DCM^{GT}, and using the property that

$\mathrm{DCM}^{GT}.\mathrm{DCM}^G = I_3$, see (*Eq. 1.3)*:

$\mathrm{DCM}^B\mathbf{r}^G = \mathrm{DCM}^B\ \mathrm{DCM}^G\mathbf{r}^B = \mathrm{DCM}^{GT}\ \mathrm{DCM}^G\mathbf{r}^B = I_3\mathbf{r}^B = \mathbf{r}^B$

Part 2. Angular Velocity

Now we have a way to characterize the orientation of one frame relative to another rotated frame, it is the DCM matrix and it allows us to easily convert the global and body coordinates back and forth using *(Eq. 1.4)* and *(Eq. 1.5).* In this section we'll analyze the rotation as a function of time that will help us establish the rules of updating the DCM matrix based on a characteristic called angular velocity. Let's consider an arbitrary rotating vector **r** and define it's coordinates at time t to be **r**(t). Now let's consider a small time interval dt and make the following notations: **r** = **r** (t) , **r'**= **r** (t+dt) and d**r = r'** – **r**:

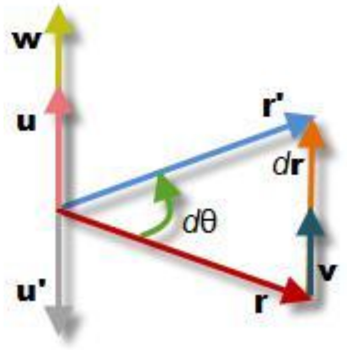


Figure 2

Let's say that during a very small time interval dt → 0 the vector **r** has rotated about an axis co-directional with a unity vector **u** by an angle dθ and ended up in the position **r'**. Since **u** is our axis of rotation it is perpendicular to the plane in which the rotation took place (the plane formed by **r** and **r**') so **u** is orthogonal to both **r** and **r**'. There are two unity vectors that are orthogonal to the plane formed by **r** and **r'**, they are shown on the picture as **u** and **u'** since we're still defining things we'llchoose the one that is co-directional with the crossproduct **r** x **r'**, following the rule of right-handed coordinate system. Thus because **u** is a unity vector |**u**| = 1 and is co-directional with **r** x **r'** we can deduct it as follows:

$\mathbf{u} = (\mathbf{r} \times \mathbf{r}') / |\mathbf{r} \times \mathbf{r}'| = (\mathbf{r} \times \mathbf{r}') / (|\mathbf{r}||\mathbf{r}'|\sin(d\theta)) = (\mathbf{r} \times \mathbf{r}') / (|\mathbf{r}|^2\sin(d\theta))$ *(Eq. 2.1)*

Since a rotation does not alter the length of a vector we used the property that| **r**'| = |**r**|.

The linear velocity of the vector **r** can be defined as the vector:

$\mathbf{v} = d\mathbf{r} / dt = (\mathbf{r}' - \mathbf{r}) / dt$ *(Eq. 2.2)*

Please note that since our dt approaches 0 so does dθ → 0, hence the angle between vectors **r** and d**r** (let's call it α) can be found from the isosceles triangle contoured by **r** , **r'** and d**r:**

$\alpha = (\pi - d\theta) / 2$ and because $d\theta \to 0$, then $\alpha \to \pi/2$

What this tells us is that **r** is perpendicular to d**r** when dt → 0 and hence **r** is perpendicular to **v** since **v** and d**r** are co-directional from *(Eq. 2.2):*

$\mathbf{v} \perp r$ *(Eq. 2.21)*

We are now ready to define the angular velocity vector. Ideally such a vector should define the rate of change of the angle θ and the axis of the rotation, so we define it as follows:

$\mathbf{w} = (d\theta/dt)\,\mathbf{u}$ *(Eq. 2.3)*

Indeed the norm of the **w** is $|\mathbf{w}| = d\theta/dt$ and the direction of **w** coincides with the axis of rotation **u**. Let's expand *(Eq. 2.3)* and try to establish a relationship with the linear velocity **v:**

Using *(Eq. 2.3)* and *(Eq. 2.1)*:

$\mathbf{w} = (d\theta/dt)\,\mathbf{u} = (d\theta/dt)(\mathbf{r} \times \mathbf{r}') / (|\mathbf{r}|^2 \sin(d\theta))$

Now note that when $dt \rightarrow 0$, so does $d\theta \rightarrow 0$ and hence for small $d\theta$, $\sin(d\theta) \approx d\theta$, we end up with:

$\mathbf{w} = (\mathbf{r} \times \mathbf{r}') / (|\mathbf{r}|^2 dt)$ *(Eq. 2.4)*

Now because $\mathbf{r}' = \mathbf{r} + d\mathbf{r}$, $d\mathbf{r}/dt = \mathbf{v}$, $\mathbf{r} \times \mathbf{r} = \mathbf{0}$ and using the distributive property of cross product over addition:

$\mathbf{w} = (\mathbf{r} \times (\mathbf{r} + d\mathbf{r})) / (|\mathbf{r}|^2 dt) = (\mathbf{r} \times \mathbf{r} + \mathbf{r} \times d\mathbf{r}) / (|\mathbf{r}|^2 dt) = \mathbf{r} \times (d\mathbf{r}/dt) / |\mathbf{r}|^2$

And finally:

$\mathbf{w} = \mathbf{r} \times \mathbf{v} / |\mathbf{r}|^2$ *(Eq. 2.5)*

This equation establishes a way to calculate angular velocity from a known linear velocity **v**.

We can easily prove the reverse equation that lets us deduct linear velocity from angular velocity:

$\mathbf{v} = \mathbf{w} \times \mathbf{r}$ *(Eq. 2.6)*

This can be proven simply by expanding **w** from *(Eq. 2.5)* and using vector triple product rule $(\mathbf{a} \times \mathbf{b}) \times \mathbf{c} = (\mathbf{a}.\mathbf{c})\mathbf{b} - (\mathbf{b}.\mathbf{c})\mathbf{a}$. Also we'll use the fact that **v** and **r** are perpendicular *(Eq.2.21)* and thus$\mathbf{v}.\mathbf{r} = 0$

$\mathbf{w} \times \mathbf{r} = (\mathbf{r} \times \mathbf{v} / |\mathbf{r}|^2) \times \mathbf{r} = (\mathbf{r} \times \mathbf{v}) \times \mathbf{r} / |\mathbf{r}|^2 = ((\mathbf{r}.\mathbf{r})\,\mathbf{v} + (\mathbf{v}.\mathbf{r})\mathbf{r}) / |\mathbf{r}|^2 = (\,|\mathbf{r}|^2\,\mathbf{v} + 0)\ |\mathbf{r}|^2 = \mathbf{v}$

So we just proved that *(Eq. 2.6)* is true. Just to check *(Eq. 2.6)* intuitively - from *Figure 2* indeed **v** has the direction of **w** x **r** using the right hand rule and indeed $\mathbf{v} \perp \mathbf{r}$ and $\mathbf{v} \perp \mathbf{w}$because it is in the same plane with **r** and **r'**.

Part 3. Gyroscopes and angular velocity vector

A 3-axis MEMS gyroscope is a device that senses rotation about 3 axes attached to the device itself (body frame). If we adopt the device's coordinate system (body's frame), and analyze some vectors attached to the earth (global frame), for example vector **K** pointing to the zenith or vector **I** pointing North - then it would appear to an observer inside the device that these vector rotate about the device center. Let w_x , w_y , w_z be the outputs of a gyroscope expressed in rad/s - the measured rotation about axes x, y , z respectively. If we query the gyroscope at regular, small time intervals **dt,** then what gyroscope output tells us is that during this time interval the earth rotated about gyroscope's *x* axis by an angle of $d\theta_x = w_x dt$, about *y* axis by an angle of $d\theta_y = w_y dt$ and about *z* axis by an angle of $d\theta_z = w_z dt$. These rotations can be characterized by the angular velocity vectors: $\mathbf{w}_x = w_x\mathbf{i} = \{w_x , 0 , 0 \}^T$, $\mathbf{w}_y = w_y\mathbf{j} = \{ 0 , w_y , 0 \}^T$, $\mathbf{w}_z = w_z\mathbf{k} = \{ 0 , 0, w_z \}^T$, where **i,j,k** are versors of the local coordinate frame (they are co-directional with body's axes x,y,z respectively). Each of these three rotations will cause a linear displacement which can be expressed by using *(Eq. 2.6)*:

$d\mathbf{r}_1 = dt\, \mathbf{v}_1 = dt\, (\mathbf{w}_x \times \mathbf{r})$; $d\mathbf{r}_2 = dt\, \mathbf{v}_2 = dt\, (\mathbf{w}_y \times \mathbf{r})$; $d\mathbf{r}_3 = dt\, \mathbf{v}_3 = dt\, (\mathbf{w}_z \times \mathbf{r})$.

The combined effect of these three displacements will be:

$d\mathbf{r} = d\mathbf{r}_1 + d\mathbf{r}_2 + d\mathbf{r}_3 = dt\, (\mathbf{w}_x \times \mathbf{r} + \mathbf{w}_y \times \mathbf{r} + \mathbf{w}_z \times \mathbf{r}) = dt\, (\mathbf{w}_x + \mathbf{w}_y + \mathbf{w}_z) \times \mathbf{r}$ (cross product is distributive over addition)

Thus the equivalent linear velocity resulting from these 3 transformations can be expressed as:

$\mathbf{v} = d\mathbf{r}/dt = (\mathbf{w}_x + \mathbf{w}_y + \mathbf{w}_z) \times \mathbf{r} = \mathbf{w} \times \mathbf{r}$, where we introduce $\mathbf{w} = \mathbf{w}_x + \mathbf{w}_y + \mathbf{w}_z = \{w_x, w_y, w_z\}$

Which looks exactly like *(Eq. 2.6)* and suggests that the combination of three **small** rotations about axes x,y,z characterized by angular rotation vectors $\mathbf{w}_x$, $\mathbf{w}_y$, $\mathbf{w}_z$ is equivalent to one **small** rotation characterized by angular rotation vector $\mathbf{w} = \mathbf{w}_x + \mathbf{w}_y + \mathbf{w}_z = \{w_x , w_y , w_z \}$.
Our main assumption that let us go from a linear displacement to a rotation by using *(Eq. 2.6)* was that dt is really small, and thus the rotations $d\theta$ and linear

displacement dr are small as well. In practice this means that the larger the dt interval between gyro queries the larger will be our accumulated error. Now, since w_x , w_y , w_z are the output of the gyroscope, then we arrive at the conclusion that in fact a 3 axis gyroscope measures the instantaneous angular velocity of the world rotating about the device's center.

Part 4. DCM complimentary filter algorithm using 9DOF IMU sensors

In the context of this text a 6DOF device is an IMU device consisting of a 3 axis gyroscope and a 3 axis accelerometer. A 9DOF device is an IMU device of a 3 axis gyroscope, a 3 axis accelerometer and a 3 axis magnetometer. Let's attach a global right-handed coordinate system to the Earth's frame such that the **I** versor points North, **K** versor points to the Zenith and thus, with these two versors fixed, the **J** versor will be constrained to point West.

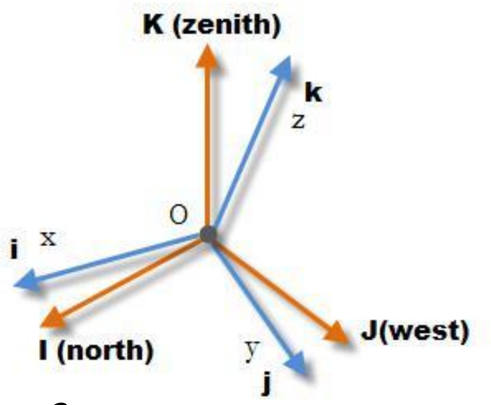


Figure 3

Also let’s consider the body coordinate system to be attached to our IMU device (acc_gyro used as an example),

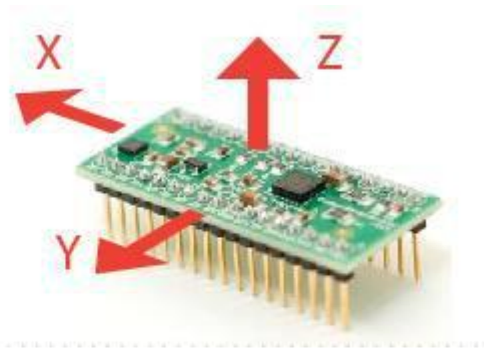


Figure 4

We already established the fact that gyroscopes can measure the angular velocity vector.

Let’s see how accelerometer and magnetometer measurements will fall into our model.

Accelerometers are devices that can sense gravitation. Gravitation vector is pointing towards the center of the earth and is opposite to the vector pointing to Zenith $\mathbf{K}^B$. If the 3 axis accelerometer output is $\mathbf{A} = \{A_x , Ay , A_z \}$ and we assume that there are no external accelerations or we have corrected them then we can estimate that $\mathbf{K}^B = -\mathbf{A}$.

Magnetometers are devices that are really similar to accelerometers, except that instead of gravitation they can sense the Earth's magnetic North. Just like accelerometers they are not perfect and often need corrections and initial calibration. If the corrected 3-axis magnetometer output is $\mathbf{M} = \{M_x , My , M_z \}$, then according to our model $\mathbf{I}^B$ is pointing North , thus $\mathbf{I}^B = \mathbf{M}$.

Knowing $\mathbf{I}^B$ and $\mathbf{K}^B$ allows us calculate $\mathbf{J}^B = \mathbf{K}^B \times \mathbf{I}^B$.

Thus an accelerometer and a magnetometer alone can give us the DCM matrix , expressed either as DCM^B or DCM^G

$DCM^G = DCM^{BT} = [\mathbf{I}^B, \mathbf{J}^B, \mathbf{K}^B]^T$

The DCM matrix can be used to convert any vector from body's(devices) coordinate system to the global coordinate system. Thus for example if we know that the nose of the plane has some fixed coordinates expressed in body's coordinate system as $\mathbf{r}^\mathbf{B} = \{1,0,0\}$, the wecan find where the device is heading in other words the coordinates of the nose in global coordinate systems using *(Eq. 1.4)*:

$\mathbf{r}^G= DCM^G \ \mathbf{r}^B$

So if an accelerometer and a magnetometer gives us the DCM matrix at any point in time, why do we need the gyroscope ? The gyroscope is actually a more precise device than the accelerometer and magnetomer are , it is used to "fine-tune" the DCM matrix returned by the accelerometer and magnetometer.

Gyroscopes have no sense of absolute orientation of the device , i.e. they don't know where north is and where zenith is (things that we can find out using the accelerometer and magnetometer), instead if we know the orientation of the device at time t, expressed as a DCM matrix DCM(t) , we can find a more precise orientation DCM(t+dt) using the gyroscope , then the one estimated directly from the accelerometer and magnetometer direct readings which are subject to a lot of noise in form of external (non-gravitational) inertial forces (i.e. acceleration) or magnetically forces that are not caused by the earth's magnetic field.

These facts call for an algorithm that would combine the readings from all three devices (accelerometer, magnetometer and gyroscope) in order to create our best guess or estimate regarding the device orientation in space (or space's orientation in device's coordinate systems), the two orientations are related since they are simply expressed using two DCM matrices that are transpose of one another ($DCM^G = DCM^{BT}$).

We'll now go ahead and introduce such an algorithm.

We'll work with the DCM matrix that consists of the versors of the global (earth's) coordinate system aligned on each row:

$$\mathrm{DCM}^G = [\mathbf{i}^G, \mathbf{j}^G, \mathbf{k}^G] = \begin{bmatrix} \mathbf{I.i} & \mathbf{I.j} & \mathbf{I.k} \\ \mathbf{J.i} & \mathbf{J.j} & \mathbf{J.k} \\ \mathbf{K.i} & \mathbf{K.j} & \mathbf{K.k} \end{bmatrix} = \begin{bmatrix} \cos(\mathbf{I},\mathbf{i}) & \cos(\mathbf{I},\mathbf{j}) & \cos(\mathbf{I},\mathbf{k}) \\ \cos(\mathbf{J},\mathbf{i}) & \cos(\mathbf{J},\mathbf{j}) & \cos(\mathbf{J},\mathbf{k}) \\ \cos(\mathbf{K},\mathbf{i}), & \cos(\mathbf{K},\mathbf{j}) & \cos(\mathbf{K},\mathbf{k}) \end{bmatrix}$$
$$= \begin{bmatrix} \mathbf{I}^{BT} \\ \mathbf{J}^{BT} \\ \mathbf{K}^{BT} \end{bmatrix}$$

If we read the rows of DCM^G we get the vectors $\mathbf{I}^B$, $\mathbf{J}^B$, $\mathbf{K}^B$. We'll work mostly with vectors $\mathbf{K}^B$(that can be directly estimated by accelerometer) and vector $\mathbf{I}^B$(that can be directlyestimated by the magnetometer). The vector $\mathbf{J}^B$ is simply calculated as $\mathbf{J}^B = \mathbf{K}^B \times \mathbf{I}^B$, since it's orthogonal to the other two vectors (remember versors are unity vectors with same direction as coordinate axes).

Let's say we know the zenith vector expressed in body frame coordinates at time t_0 and we note it as $\mathbf{K}^B_{0.}$ Also let's say we measured our gyro output and we have determined that our angular velocity is $\mathbf{w} = \{w_x , w_y , w_z \}$. Using our gyro we want to know the position of our zenith vector after a small period of time dt has passed we'll note it as $\mathbf{K}^B_{1G}$. And we find it using *(Eq. 2.6)*:

$\mathbf{K}^B{}_{1G} \approx \mathbf{K}^B{}_0 + dt\ \mathbf{v} = \mathbf{K}^B{}_0 + dt\ (\mathbf{w_g} \times \mathbf{K}^B{}_0) = \mathbf{K}^B{}_0 + (\ d\boldsymbol{\theta}_\mathbf{g} \times \mathbf{K}^B{}_0)$

Where we noted $d\boldsymbol{\theta}_\mathbf{g} = dt\ \mathbf{w_g}$. Because $\mathbf{w_g}$ is angular velocity as measured by the gyroscope. We'll call $d\boldsymbol{\theta}_\mathbf{g}$ angular displacement. In other words it tells us by what **small** angle (given for all 3 axis in form of a vector) has the orientation of a vector $\mathbf{K}^B$ changed during this **small** period of time dt.

Obviously, another way to estimate $\mathbf{K}^B$ is by making another reading from accelerometer so we can get a reading that we note as $\mathbf{K}^B{}_{1A}$.

In practice the values $\mathbf{K}^B{}_{1G}$ will be different from from $\mathbf{K}^B{}_{1A.}$ One was estimated using our gyroscope and the other was estimated using our accelerometer.

Now it turns out we can go the reverse way and estimate the angular velocity $\mathbf{w}_a$ or angular displacement $d\boldsymbol{\theta}_\mathbf{a} = dt\ \mathbf{w_a}$, from the new accelerometer reading $\mathbf{K}^B{}_{1A}$, we'll use *(Eq. 2.5):*

$\mathbf{w}_a = \mathbf{K}^B{}_0 \times \mathbf{v_a} / \mid \mathbf{K}^B{}_0 \mid^2$

Now $\mathbf{v}_a = (\mathbf{K}^B{}_{1A} - \mathbf{K}^B{}_0) / dt$, and is basically the linear velocity of the vector $\mathbf{K}^B{}_0$. And $\mid \mathbf{K}^B{}_0 \mid^2 = 1$, since $\mathbf{K}^B{}_0$ is a unity vector. So we can calculate:

$d\boldsymbol{\theta}_\mathbf{a} = dt\ \mathbf{w_a} = \mathbf{K}^B{}_0 \times (\mathbf{K}^B{}_{1A} - \mathbf{K}^B{}_0)$

The idea of calculating a new estimate $\mathbf{K}^B{}_1$ that combines both $\mathbf{K}^B{}_{1A}$ and $\mathbf{K}^B{}_{1G}$ is to first estimate $d\boldsymbol{\theta}$ as a weighted average of $d\boldsymbol{\theta}_\mathbf{a}$ and $d\boldsymbol{\theta}_\mathbf{g}$:

$d\boldsymbol{\theta} = (s_a\ d\boldsymbol{\theta}_\mathbf{a} + s_g\ d\boldsymbol{\theta}_\mathbf{g}) / (s_a + s_g)$, we'll discuss about the weights later on , but shortly they are determined and tuned experimentally in order to achieve a desired response rate and noise rejection.

And then $\mathbf{K}^B{}_1$ is calculated similar to how we calculated $\mathbf{K}^B{}_{1G}$:
$\mathbf{K}^B{}_1 \approx \mathbf{K}^B{}_0 + (\ d\boldsymbol{\theta} \times \mathbf{K}^B{}_0)$

We went all the way to calculate $d\boldsymbol{\theta}$ and did not apply the weighted average formula directly to $\mathbf{K}^B{}_{1A}$ and $\mathbf{K}^B{}_{1G}$ because $d\boldsymbol{\theta}$ can be used to calculate the other elements of our DCM matrix in the same way:

$\mathbf{I}^B{}_1 \approx \mathbf{I}^B{}_0 + (d\boldsymbol{\theta} \times \mathbf{I}^B{}_0)$

$\mathbf{J}^B{}_1 \approx \mathbf{J}^B{}_0 + (d\boldsymbol{\theta} \times \mathbf{J}^B{}_0)$

The idea is that all three versors $\mathbf{I}^B$, $\mathbf{J}^B$, $\mathbf{K}^B$ are attached to each other and will follow the same angular displacement $d\boldsymbol{\theta}$ during our small interval dt. So in a nutshell this is the algorithm that allows us to calculate the DCM_1 matrix at time t_1 from our previous estimated DCM_0 matrix at time t_0. It is applied recursively at regular small time intervals dt and gives us an updated DCM matrix at any point in time. The matrix will not drift too much because it is fixed to the absolute position dictated by the accelerometer and will not be too noisy from external accelerations because we also use the gyroscope data to update it.

So far we didn't mention a word about our magnetometer and we can go away without using it, but our resulting orientation will then have a drifting heading (i.e. it will not show if we're heading north, south, west or east), or we can introduce a virtual magnetometer that is always pointing North, to introduce stability in our model.

Now we'll show how to integrate magnetometer readings into our algorithm. As it turns out it is really simple since magnetometer is really similar to accelerometer (they even use similar calibration algorithms), the only difference being that instead of estimating the Zenith vector $\mathbf{K}^B$ vector it estimates the vector pointing North $\mathbf{I}^B$. Following the same logic as we did for our accelerometer we can determine the angular displacement according to the updated magnetometer reading as being:

$d\boldsymbol{\theta}_\mathbf{m} = dt\ \mathbf{w}_\mathbf{m} = \mathbf{I}^B{}_0 \times (\mathbf{I}^B{}_{1M} - \mathbf{I}^B{}_0)$

Now let's incorporate it into our weighted average:

$d\boldsymbol{\theta} = (s_a\ d\boldsymbol{\theta}_\mathbf{a} + s_g\ d\boldsymbol{\theta}_\mathbf{g} + s_m\ d\boldsymbol{\theta}_\mathbf{m}) / (s_a + s_g + s_m)$

From here we go the same path to calculate the updated DCM_1

$\mathbf{I}^B{}_1 \approx \mathbf{I}^B{}_0 + (\, d\boldsymbol{\theta} \times \mathbf{I}^B{}_0)$, $\mathbf{K}^B{}_1 \approx \mathbf{K}^B{}_0 + (\, d\boldsymbol{\theta} \times \mathbf{K}^B{}_0)$ and $\mathbf{J}^B{}_1 \approx \mathbf{J}^B{}_0 + (\, d\boldsymbol{\theta} \times \mathbf{J}^B{}_0)$

ALGORITHM:

The algorithm for the controller is as follows:

Step1: Initialize the values of IMU bias and accelerometer bias for zero value of θ & ω.

Step2: Measure the value of voltage of gyro and accelerometer outputs and store those values as ω and Accelerations.

Step3: Integrate the IMU reading by: $\theta_t = \theta_{t-1} + \omega^*\delta t$
Differentiate the IMU reading by: $\acute{\omega}_t = (\omega_t - \omega_{t-1})/\ \delta t$
δt in each case is assumed to be constant and is equal to the cycle time.

Step4: Calculate the raw angle and gravity angle and compute the error value.

Step5: Update the raw angle by:
$\theta_{\text{stabilized}} = \theta_{\text{raw angle}} + \text{constant}^*\ (\theta_{\text{gravity angle}} - \theta_{\text{raw angle}})$

Step6: We calculate the torque required to restore balance:
$\tau = \text{constant} * \sin\theta$ (or $\tau = \text{constant} * \theta$ for small angles)

Step7: We obtain the direction of rotation of the reaction wheel:
If $\theta < 0$, counter-clockwise
If $\theta > 0$, clockwise

Step8: The PWM provided to the motor through Arduino is given by

$$\text{PWM} = \text{constant} * \theta^2$$

Step9: Repeat steps (2-8)