```
        x = np.copy(xtraj[:,i])
    return xtraj
```

Problem 1 (20pts)

Show that if $R \in SO(n)$, then the matrix $A = \frac{d}{dt}(R)R^{-1}$ is skew symmetric.

Turn in: A scanned (or photograph from your phone or webcam) copy of your hand written solution. Or you can use \LaTeX.

In [ ]:

Problem 2 (20pts)

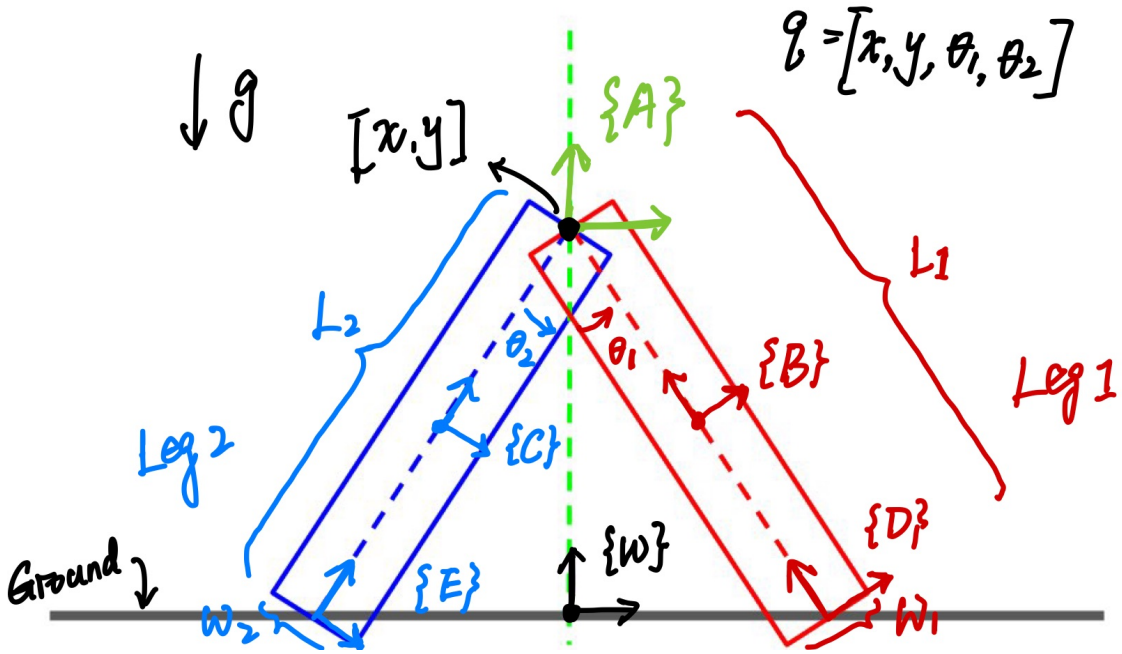
Show that $\underline{\widehat{\omega}}\ \underline{r}_b = -\underline{\hat{r}}_b\ \underline{\omega}$.

Turn in: A scanned (or photograph from your phone or webcam) copy of your hand written solution. Or you can use \LaTeX.

In [ ]:

In [27]:

```
from IPython.core.display import HTML
display(HTML("<table><tr><td><img src='https://github.com/MuchenSun/ME314pngs/raw/master/bi
```



Problem 3 (60pts)

Consider a person doing the splits (shown in the image above). To simplify the model, we ignore the upper body and assume the knees can not bend --- which means we only need four variables $q = [x, y, \theta_1, \theta_2]$ to configure the system. x and y are the position of the intersection point of the two legs, θ_1 and θ_2 are the angles between the legs and the green vertical dash line. The feet are constrained on the ground, and there is no friction between the feet and the ground.

Each leg is a rigid body with length $L = 1$, width $W = 0.2$, mass $m = 1$, and rotational inertia $J = 1$ (assuming the center of mass is at the center of geometry). Moreover, there are two torques applied on θ_1 and θ_2 to control the legs to track a desired trajectory:

$$\theta_1^d(t) = \frac{\pi}{15} + \frac{\pi}{3}\sin^2(\frac{t}{2})$$
$$\theta_2^d(t) = -\frac{\pi}{15} - \frac{\pi}{3}\sin^2(\frac{t}{2})$$

and the torques are:

$$F_{\theta_1} = -k_1(\theta_1 - \theta_1^d)$$
$$F_{\theta_2} = -k_1(\theta_2 - \theta_2^d)$$

In this problem we use $k_1 = 20$.

Given the model description above, define the frames that you need (several example frames are shown in the image as well), simulate the motion of the biped from rest for $t \in [0, 10], dt = 0.01$, with initial condition $q = [0, L_1 \cos(\frac{\pi}{15}), \frac{\pi}{15}, -\frac{\pi}{15}]$. You will need to modify the animation function to display the leg as a rectangle, an example of the animation can be found at https://youtu.be/w8XHYrEoWTc (https://youtu.be/w8XHYrEoWTc).

Hint 1: Even though this is a 2D system, in order to compute kinetic energy from both translation and rotation you will need to model the system in the 3D world --- the z coordinate is always zero and the rotation is around the z axis (based on these facts, what should the SE(3) matrix and rotational inertia tensor look like?). This also means you need to represent transformations in SE(3) and the body velocity $\mathcal{V}_b \in \mathbb{R}^6$.

Hint 2: It could be helpful to define several helper functions for all the matrix operations you will need to use. For example, a function that returns SE(3) matrices given a rotation angle and 2D translation vector, functions for "hat" and "unhat" operations, a function for the matrix inverse of SE(3) (which is definitely not the same as the SymPy matrix inverse function), and a function that returns the time derivative of SO(3) or SE(3).

Hint 3: In this problem the external force depends on time t. Therefore, in order to solve for the symbolic solution you need to substitute your configuration variables, which are defined as symbolic functions of time t (such as $\theta_1(t)$ and $\frac{d}{dt}\theta_1(t)$), with dummy symbolic variables. For the same reason (the dynamics now explicitly depend on time), you will need to do some tiny modifications on the "integrate" and "simulate" functions, a good reference can be found at https://en.wikipedia.org/wiki/Runge-Kutta_methods (https://en.wikipedia.org/wiki/Runge-Kutta_methods).

Hint 4: Symbolically solving this system should be fast, but if you encountered some problem when solving the dynamics symbolically, an alternative method is to solve the system numerically --- substitute in the system state at each time step during simulation and solve for the numerical solution --- but based on my experience, this would cost more than one hour for 500 time steps, so it's not recommended.

Hint 5: The animation of this problem is similar to the one in last homework --- the coordinates of the vertices in the body frame are constant, you just need to transfer them back to the world frame using the the transformation matrices you already have in the simulation.

Hint 6: Be careful to consider the relationship between the frames and to not build in any implicit assumptions (such as assuming some variables are fixed).

Hint 7: The rotation, by convention, is assumed to follow the right hand rule, which means the z-axis is out of the screen and the positive rotation orientation is counter-clockwise. Make sure you follow a consistent set of positive directions for all the computation.

Hint 8: This problem is designed as a "mini-project", it could help you estimate the complexity of your final project, and you could adjust your proposal based on your experience with this problem.

Turn in: A copy of the code used to simulate and animate the system. Also, include a plot of the trajectory and upload a video of the animation separately through Canvas. The video should be in ".mp4" format, you can use screen capture or record the screen directly with your phone.

In [16]:

```
## Your code goes here
```