1. Gravity is set to -9.81 ms-2 in the z-axis. No other forces acting. Friction has been ignored for now.
2. The state at each time step is defined by a list of length 11. A state consists of the following physical parameters: [x, y, vx, vy, vz, ux, uy, uz, roll, pitch, yaw].
   1. x 🡪 x co-ordinate of two wheeler
   2. y 🡪 y co-ordinate of two wheeler
   3. vx, vy, vz,🡪 linear velocity along 3 axes.
   4. ux, uy, uz 🡪 angular velocity along 3 axes
3. The action corresponding to a particular state space is a list of length 3. An action consists of [momentum wheel velocity, back wheel velocity, steering angle]
4. The reward function calculates the total reward for a state-action pair.
   1. Distance to destination is calculated using the Euclidean formula.
      1. Distance = sqrt ((x – dx)2 + (y - dy)2)
      2. Obs: The negative of this value is finally added to the reward function in order to encourage two wheeler’s movement toward the goal.
   2. A reward is given for being upright. This value is calculated as follows:
      1. upright\_reward = 1.0 − ( abs(roll) + abs(pitch) )
      2. The closer the two wheeler is to being perfectly perpendicular to the plane, more the reward.
      3. Pitch is generally 0. Could probably eliminate it from the algorithm.
      4. Obs: This reward never exceeds 1. Compared to the distance reward, this seems miniscule and thus might not contribute much to toal reward.
   3. A general positive velocity is also rewarded as we have set the destination to always be towards the positive x and y axes. For reference, the two wheeler always spawns at the origin.
      1. velocity\_reward = max (0, linear\_velocity)
      2. The model gets 0 reward if it is moving backward.
      3. Obs: Discovered a bug. The code only considers the x axis linear velocity and ignores the y axis for calculating reward.
5. The total reward is the sum of all the smaller rewards stated above.
6. An episode concludes if either the two-wheeler reaches its destination or it completely tips over.
7. While training the second version, we also observed that the model was exploring and exploiting reasonable well in the initial episodes. Eventually, when we crossed 35 episodes, the model started exploiting more. By episode 50, the model was almost exclusive exploiting the environment. Here, we observed that every episode concluded almost as soon as the two-wheeler spawns. The model could not stabilize the two wheeler and gave a very large input to the momentum wheel which caused the two-wheeler to tip over almost instantly.
   1. This could possibly be caused by the use of a total reward as opposed to individual rewards for each action.