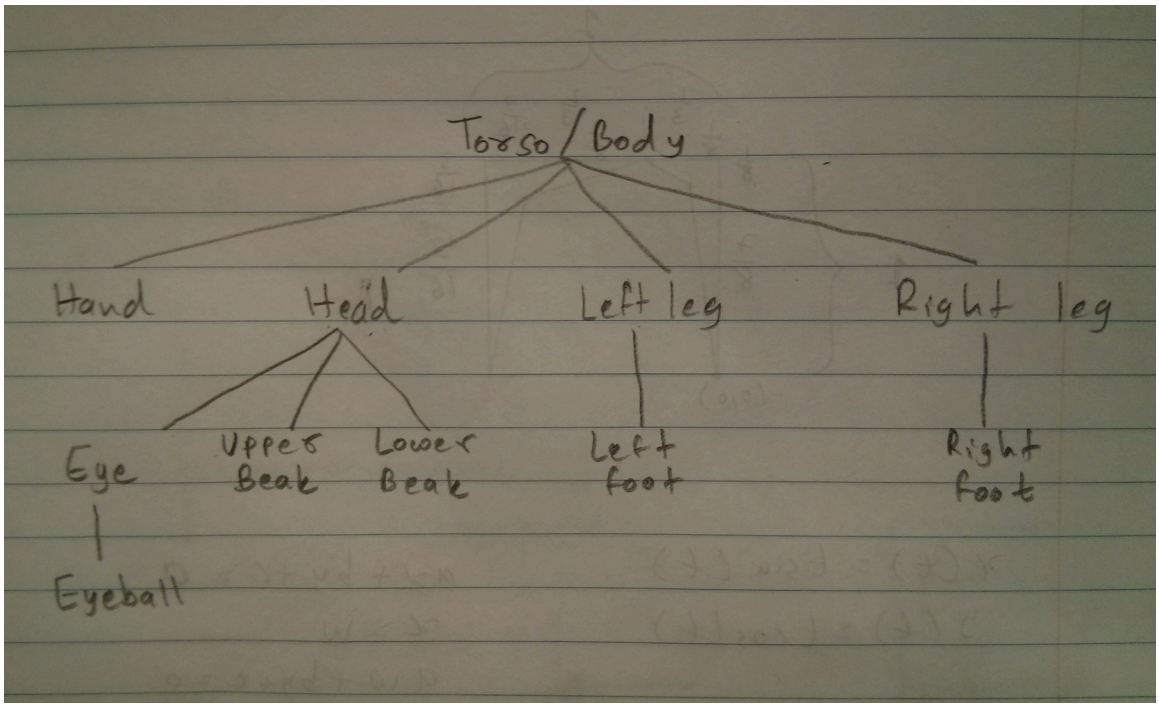
CSC418
Assignment 1

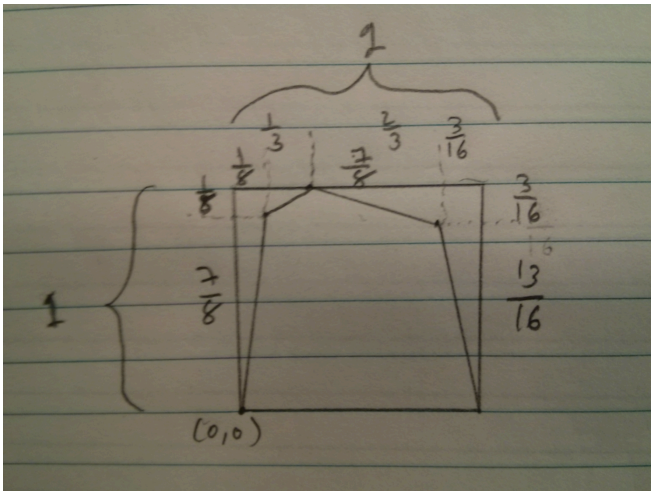
Design

To design the Penguin, I used Hierarchical Kinematic Modeling, since it allows us to easily perform transformations and ensure that the object remains connected. It also applies the transformation on all child nodes, which makes it easy to work with. I used the torso as the root node of the design and built around it in a Kinematic Modeling method.



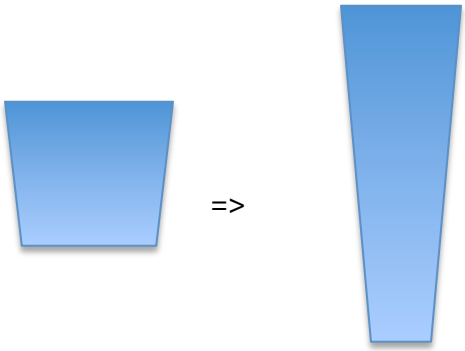
The method I used for drawing parts was same as the one in example. I draw the parts using 1x1 dimensions and than scale it to match the X and Y coordinates to the actual shape's width and length.

Below is an example of how I worked on the head of the Penguin. I connected points (0, 0), (1/8, 7/8), (1/3, 1), (13/16, 13/16), (1,0) to draw a unit polygon. I used fractions to draw most of my shapes. I shifted the head back to (-0.5, -0.5) so that its center is (0,0).



For parts like Arm, Legs, Feet and Beak, I used *drawTrapezoid* function to draw a basic trapezoid and than used scaling.

For Hand, I drew a basic trapezoid with width1 > width2 and scaled it to make it larger. I designed the legs and feet the same way:



Then I used rotation to fix legs and feet to proper direction.

To solve the problem of having to use very small dimensions in child objects, I used

glScalef(1/parent_height, 1/parent_width, 1.0);

Parts

Torso

- I designed a dedicated function *drawBody* to draw torso. It draws 6 points in 1x1 area. I scale it later to make it fit to desired height and width.

Hand

- I draw Arm as a parent of the Torso.

- To draw Arm, I draw trapezoid and move it to center, make it larger by using scale and assign it *hand_rot* for the movement that goes from -45 to 45 degrees.
- I draw the joint circle and scale it back to (1/TORSO_WIDTH, 1/TORSO_LENGTH).

Legs
- I draw legs using *drawTrapezoid* function as a unit trapezoid and than I scale it to make it larger. I use *joint_rot* and *joint_rot_inv* to rotate left leg and right leg respectively.
- I draw legs as child of the Torso and give feet as it's children.

Feet
- I draw feet the same way I draw legs, by using trapezoid and scaling it.
- I attach feet to legs and give it rotation in range -10 to 10 degrees.

Head
- I draw head as a child of Torso as well. I use dedicated function *drawHead* to draw 5 vertices for head.
- I then scale it to be of bigger size and give it -8 to 8 degrees of freedom.
- I shift up the head so th

Beak
- I draw beak using *drawTrapezoid* function. I scale the beak and than rotate it 90 degrees.

- I translate upper beak higher and give lower beak a rotation transformation.

Eye
- I draw eyes as child of the Head as well.
- I simply draw circle without fill for the eye.
- I also have eyeball as child of eyes that uses smaller circle with fill. And translates bit to left-down.

Road
- This is just extra polygon I drew so that it is easier to visualize even the minor transformations.

Transformations

I used Rotation, Translation and Scale transformations for this project

Torso

- The Torso translates on X, Y axis.
- It goes from -400 to 400 on X-axis and goes back to -400 as *animation_frame* resets to 0.
- To move the torso on Y-axis, I use *abs(head_rot/2)* so that it goes up and down from (8/2 =) 4 to 0 degrees.
- I do *400 - animation_frame*2* for X-axis movement so it goes at fixed speed.

Hand

- Hand rotates -45 to 45 degrees through it's joint.
- I use the given *joint_rot* to rotate the hand.

Legs

- Legs also rotate -45 degrees to 45 degrees.
- To keep it simple and concise, I have the left leg rotating as the inverse of the Arm *(-joint_rot)* and right leg moving with the same angle as the Arm.

Feet

- For right foot, I use the same sin function as Arm and find angle between range -10 and 10 degrees and rotate accordingly.
- For the left foot, I just use the negative of right foot for the rotation.

Head

- For the head rotation, I use the same sin function as Arm but I have it from range -8 to 8 degrees.

Beak

- The upper beak does not move.
- For the lower beak, I simply use translation of *(head_rot+HEAD_JOINT_MIN),* which is from range -16 to 0.
- Thus it goes down by -16 each step and translates back to 0.