Table of Contents

Abstract

A walking six-legged robot using prismatic and revolute joints simultaneously is created on ADAMS 17.2 Student Edition simulation software. This robot measures approximately 3cm x 1cm x 4cm and has the ability to travel 3cm in 10 seconds in a straight line. The following report documents the creation of this robot and provides charts and snapshots of the robot in motion.

Instructions

The following are instructions on how to recreate the six-legged walking robot. All measurements are in millimeters (mm). To set up the model, open the software and select New Model and the units 'MMKS – mm, kg, N, s, deg'.

Step 1: Create the Structure

Ground

Under the tab "Bodies", click on the box icon in the 'Solids' section. On the left panel, use the drop-down box and select "On Ground". Check the boxes next to Length, Height, and Depth, and enter 1000.0, 30.0, and 300.0 respectively. To place the box in the working environment, hover your mouse over the blue area to its desired location and click.

Robot

Body

The body of the robot can be created by the same procedure as how the ground was constructed. However, when defining the box dimensions, enter 250, 50, and 100 for its Length, Height, and Depth, respectively.

Legs

The legs are created by using two links, where one upper link is connected to the body with a revolute joint, and the other lower link is connected to its respective upper link with a prismatic joint.

Under the tab "Bodies", click on the link icon and then click on any lower corner of the robot body. Bring the mouse down and then click to create a perpendicular link to the body. Add the lower link in the same fashion, but this time start the creation at the midpoint of the upper link and extend its endpoint past the upper. These two links should have the dimensions of 10, 5, and 450 for its Width, Depth, and Length, respectively.

Now that one leg has been created, copy these two parts by selecting their name on the tree, and then right click to bring up a popup and select copy. Repeat this until six complete legs, with upper and lower links, have been created.

Since the copies of the legs have been placed in the exact location as the original, you will need to reposition them to the other corners and sides of the robot body. On each link, there is one coordinate (named MARKER_## by default) that can move the part if the X, Y, or Z values are altered. It will take a few tries to figure out which coordinate it is, as there are several of them per link, and find the values that will position them in the correct locations. There should be four legs at the lower corners of the robot body, and two legs on the midpoint of the longest length.

Step 2: Assign Connections

To secure the legs to the body and establish a ground for the robot to walk on, three types of connectors are necessary: fixed, revolute, and prismatic joints.

Fixed Joints for Locking the Ground

Under the tab "Connectors", select the lock icon, and then select the ground followed by any space in the blue area. Click on any edge of the ground to place the lock.

Revolute Joint for Legs

Revolute joints connect the upper link to the body of the robot. In the same tab, select the icon for revolute joint, and then click on an upper link followed by the robot body. Click on the midpoint at the end of the link that is closest to the body to place the joint. Repeat for all five other upper links.

Prismatic Joint for Legs

Prismatic joints connect the lower link to the upper link and allows it to slide linearly with respect to each other. In the same tab, select the icon for prismatic joint, and then click on the lower link followed by the upper link. Click on the midpoint of the lower link to place the prismatic joint, and then click directly below it to assign a perpendicular motion to the ground. Repeat for all 5 other lower links.

Step 3: Add Joint Motions

Since the direction constraints have been now placed at the joints, it is crucial to next apply the motions to them. This will add further constraints as to how far the parts and rotate or translate, and will help propel the robot in a walking manner.

Revolute Motion

Under the "Motions" tab, click on the rotational joint motion icon, and then click on any revolute joint. Repeat for the other five revolute joints.

Navigate to the tree to where it lists all the motions that have been created. Identify which legs are the "A" legs and which are the "B" legs – see Figure 1 – and then rename it in the tree accordingly; this will be helpful for the future steps.

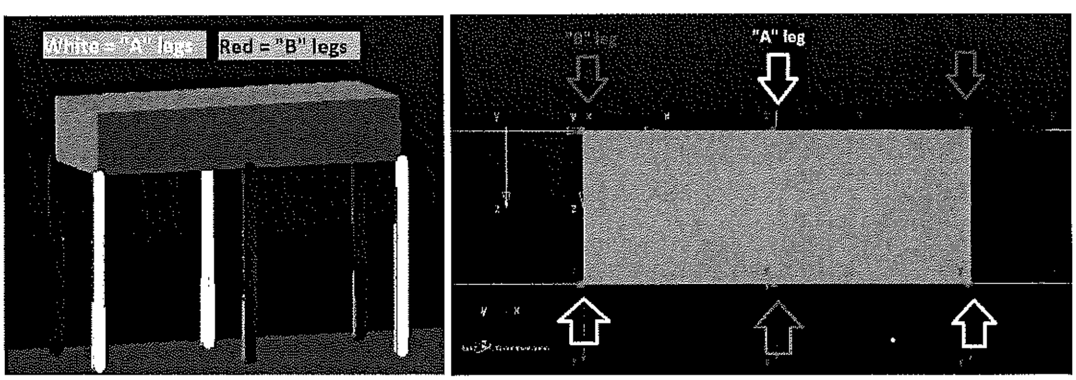

Figure 1 Identification of A and B Legs. Isometric view (left), top view (right)

Double click on a revolute joint motion item belonging to any "A" leg to bring up a pop up window. Click on the button next to the line "Function (time)" to bring up the Function Builder. In the gray box under 'Define a runtime function", enter the following:

```
STEP(time, 0.0, 0.0, 1.0, -0.4) + STEP(time, 1.2, 0.0, 2.2, 0.4) + STEP(time, 3.2, 0.0, 4.2, -0.4) + STEP(time, 4.2, 0.0, 5.2, 0.4) + STEP(time, 6.2, 0.0, 7.2, -0.4) + STEP(time, 7.2, 0.0, 8.2, 0.4)
```

Repeat for all the joints for the "A" legs.

For all the revolute joint motions of the "B" legs, repeat the step but this time enter the following:

```
STEP(time, 1.8, 0.0, 2.8, -.40)+STEP(time, 2.8, 0.0, 3.8, 0.4)+STEP(time, 4.8, 0.0, 5.8, -0.4)+STEP(time, 5.8, 0.0, 6.8, 0.4)+STEP(time, 7.8, 0.0, 8.8, -0.4)+STEP(time, 8.8, 0.0, 9.8, 0.4)
```

Make sure to click "OK" or "apply" for every window to ensure that all functions have been successfully added.

Prismatic Motion

Under the "Motions" tab, click on the translational joint motion icon, and then click on any translational joint. Repeat for the other five translational joints.

Navigate to the tree to where it lists all the motions that have been created. This time, find all the translational joints related to the "A" legs and enter the following into the functions builder:

```
STEP(time, 0.60, 0.0, 1.2, -13.0)+STEP(time, 2, 0.0, 2.6, 13.0)+STEP(time, 4, 0.0, 4.6, -13.0)+STEP(time, 5, 0.0, 5.6, 13.0)+STEP(time,7, 0.0, 7.6, -13.0)+STEP(time,8, 0.0, 8.6, 13.0)
```

For all the translational joints related to the "B" legs, enter the following in the functions builder:

```
STEP(time, 0.0, 0.0, 1.0, 0.0)+STEP(time, 2.2, 0.0, 2.8, -13.0)+STEP(time, 3.6, 0.0, 4.2, 13.0)+STEP(time, 5.2, 0.0, 5.8, -13.0)+STEP(time, 6.6 ,0.0, 7.2, 13.0)+STEP(time, 8.2 ,0.0, 8.8, -13.0)
```

Step 4: Assign Forces (Surface Contacts)

When running the simulation on ADAMS, it is necessary to create surface contacts that will prevent the objects from falling endlessly into space. Assigning a surface contact with all the lower links to the ground is sufficient to keep the robot on top of the ground.

Under the "Forces" tab, click on the 'Create a Contact' icon. In the 'I Solid(s)' line, right click and then click Contact_Solid > pick, and then select one of the six lower links. In the 'J Solid(s)' line, rick click then then click Contact_Solid > pick, and then select the ground. Click OK. Repeat for all the other five lower links.

Step 5: Simulate

Under the "Simulation" tab, click on the gear icon to, and define the End Time and Steps as 10 and 500, respectively. Click on the green triangle to simulate the robot.

A visualization of the robot during its construction and while it's in motion is given in Figure 2.

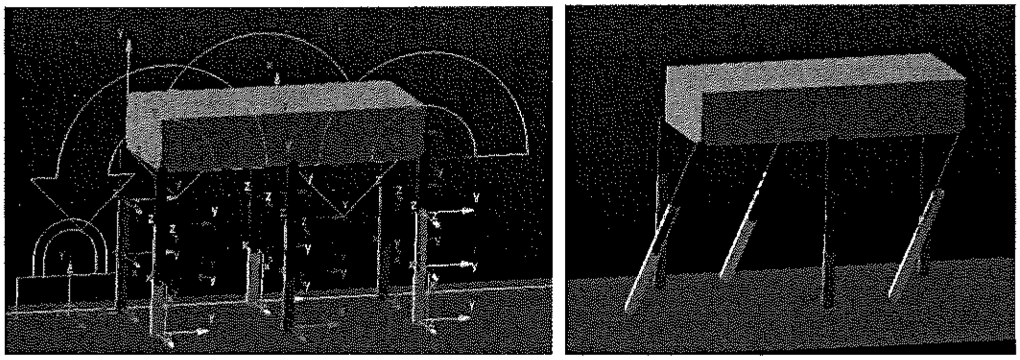

Figure 2 Constructing the Robot (Left) and Robot in Motion(Right)

Helpful Tips

- Keyboard shortcuts: (For viewing) 't' to translate, 'r' to rotate, 'z'+ click and drag to zoom in/out. (For views) Shift + 't' for top, or 'f' for front, or 'r' for right views.
- When selecting icons that will create solid bodies, connectors, motions or forces, there may be a message at the lower corner of the ADAMS window that gives instructions on what steps are necessary to take to complete the action.
- Oftentimes, the working space can get crowded with coordinates/bodies and it becomes difficult to select or view them. Deactivating or changing the visibility of the coordinate or solid body in the tree will help clean up the space.

Results

The robot's is designed to propel itself in the following manner and is shown in Figure 3. From the initial position in the snapshot (1) to snapshot (3), "A" legs (white) first rotates forward to about 20° and extend via its prismatic joint to touch the ground. A rotation force which can be supplied by a motor is applied to all "A" legs' revolute joints and propels the robot forward as seen in snapshots (4) to (5). While doing so, the "A" legs simultaneously retract back to its original length. "B" legs (green) remain rigid and vertical as a stand to support the robot body until the "A" legs have returned to its vertical position. Once so, "B" legs rotate forward to 20° and extend via its prismatic joint to touch the ground, and repeats in the same process as "A". When "A" and "B" takes turn moving in this fashion, the robot travels forward.

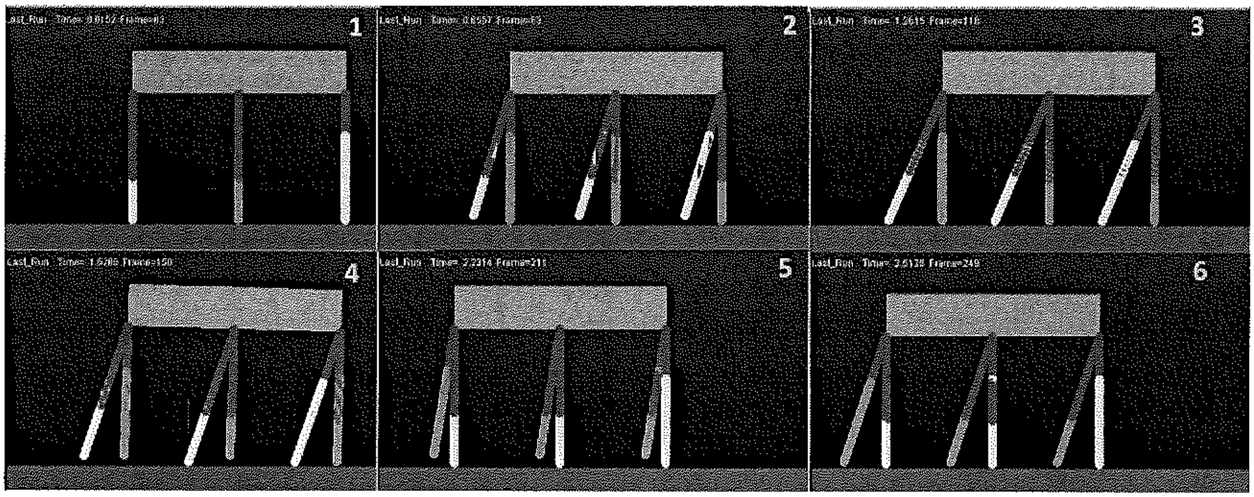

Figure 3 Side-view Snapshots in Time of the Six-Legged Robot Walking

Step functions were used on the prismatic and revolute joint motions to simulate the robot walking. All the values that are used in the step functions to define the initial time and position and final time and position for all "A" and "B" legs are noted below in Figure 4. In general, all rotations are given a duration of 1 second to complete, and all translations are allowed 0.6 seconds to displace. The final position for all rotations alternate between 0.4 or -0.4, and the final position for all translations are either -13 or 13. All initial positions are relative to the last move's final position. It is realized that from an even-numbered move to the next odd-numbered move, there is a 2 second difference in initial times, and from an odd-numbered move to the next even-numbered move, there is a 1 second difference in intial times (with the exception for B Translations).

	Move1				Move2				Move3				Move4				Move5				Move6			
	init time	init pos	final time	final pos	init time	init pos	final time	final pos	init time	init pos	final time	final pos	init time	init pos	final time	final pos	init time	init pos	final time	final pos	init time	init pos	final time	final pos
A Rotation	0	0	1	-0.4	1.2	0	2.2	0.4	3.2	0	4.2	-0.4	4.2	0	5.2	0.4	6.2	0	7.2	-0.4	7.2	0	8.2	0.4
A Translation	0.6	0	1.2	-13	2	0	2.6	13	4	0	4.6	-13	5	0	5.6	-13	7	0	7.6	-13	8	0	8.6	-13
B Rotation	1.8	0	2.8	-0.4	2.8	0	3.8	0.4	4.8	0	5.8	-0.4	5.8	0	6.8	0.4	7.8	0	8.8	-0.4	8.8	0	9.8	0.4
B Translation	0	0	0.6	-13	2.2	0	2.8	-13	3.6	0	4.2	-13	5.2	0	5.8	-13	6.6	0	7.2	-13	8.2	0	7.8	-13

Figure 4 Step Function Values

The graph below plots the position of coordinate of the robot's body's center of mass. From the X-position of the coordinate, is observed that it can travel a little more than 300mm in a 10 second time frame, and walks fairly straight in the X-direction since the Y and Z position of the coordinate is relatively unchanging. Observing the X-position curve, it can be concluded that the robot spends more time resting than traveling, as indicated in how there are longer plateaus than dips.

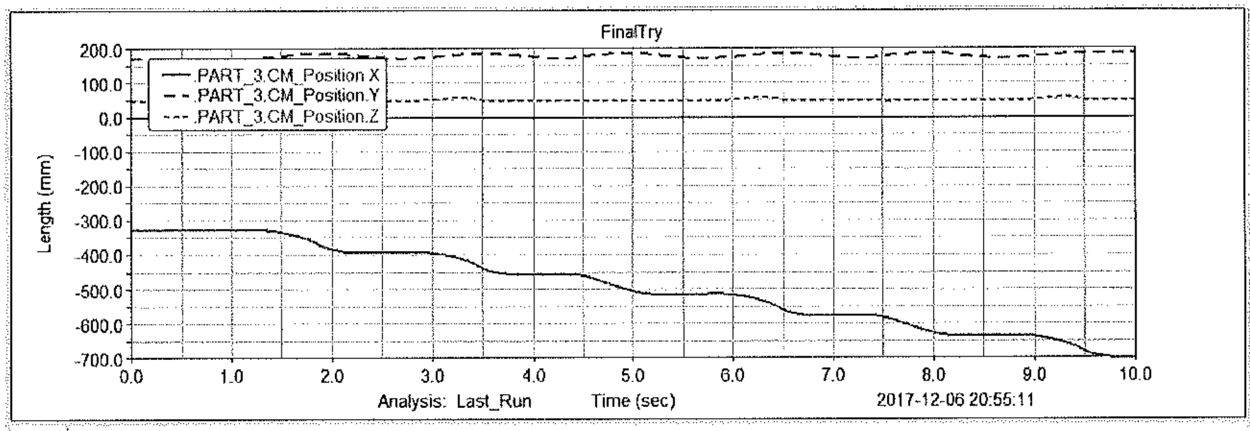

Figure 5 Robot Body's Position Graph

Conclusion

This robot is rather simple to build in such that it really only requires three different shapes – a box for the body, links for the leg, and another box for the ground. Because the geometry is elementary, it was easy to adjust the coordinates so that the links were positioned in the right location. Because the robot simultaneously requires the legs extend while rotating to rebalance itself, the difficulty lied in finding the optimal values for the revolute and prismatic joint functions so that it can operate smoothly. Much trial and error was done to achieve the values shown in Figure 4, but once a few moves were established, a pattern is realized and can be easily applied to create additional moves. Possible modifications to the robot may include altering the "feet" of the robot, that is, in how the leg comes into contact with the surface of the ground. Perhaps a small and flat box can be merged to the end of each leg so that the contact area is larger for robot to be improved in its walking; this situation is likely to be more realistic than having only one line of contact (since currently the robot's "feet" is curved). The step function values can also be further altered to provide a smoother travel motion for the robot.