Detailed MBSim Refrences Section:

USING MBSim: SCRIPT FILES

The MBSim user describes a new mechanism through script files. These are simple text files that are read by MBSim during program execution. In this way a vehicle can be built up or modified without having to recompile the source code. Thus the main key to using MBSim is mastering these script files. Currently script files are used to describe the mechanism's joints, mass properties, physical appearance, sensor type and placement, initial configuration, joint velocities, and environment.

One creates and uses a new simulation in the following manner. First one must set up script files that describe the mechanism, sensors, initial configuration, velocity, environment, and the graphics view. Then the main() should be written. This code reads the script files, drives the mechanism, and activates the sensors (See SAMPLE PROGRAM).

All dimensions are in millimeters, angles in radians, mass in grams, and time in seconds. Another consistent set of dimensions can be substituted. Commas should not be used between values.

DESCRIBING A MECHANISM

A mechanism is defined with a mechanism description file. In MBSim, each mechanism is constructed as a series of links. And the user defines the parameters, derived from Denavit–Hartenberg convention [2][3], for each link and its associated joint. Note that a link is currently assumed to have only one joint. These parameters are followed by a graphics description filename and an optional sensor description filename and a list of sensor identification numbers. These numbers will be used to activate the sensors they refer to.

Mechanism description file elements are now described.

==========

Mechanism Description Files (.md) (Describes and builds Mechanism)

The first line contains the name of the mechanism. Then the list of joints and their parameters. Available joints are:

Revolute Joint (R): alpha, a, d, theta (D–H parameters Fig. 2)
Standard rotary joint found on most industrial robots.

Prismatic Joint (P): alpha, a, d, theta (D–H parameters Fig. 3)
Telescopic joint.

Unactuated Joint (U): alpha, a, d, theta (D–H parameters)
Often used to represent a tool in a manipulator's gripper.

Knife Joint (K): x,y,z, theta
Can be thought of (with Dual Wheel and Tricycle) as a joint between the mechanism and the ground. moves similar to a pizza cutter:

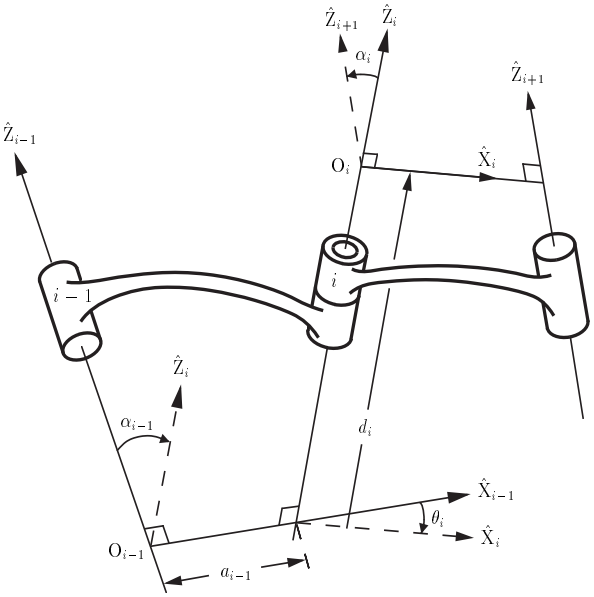

Figure 2 – Revolute joint parameters

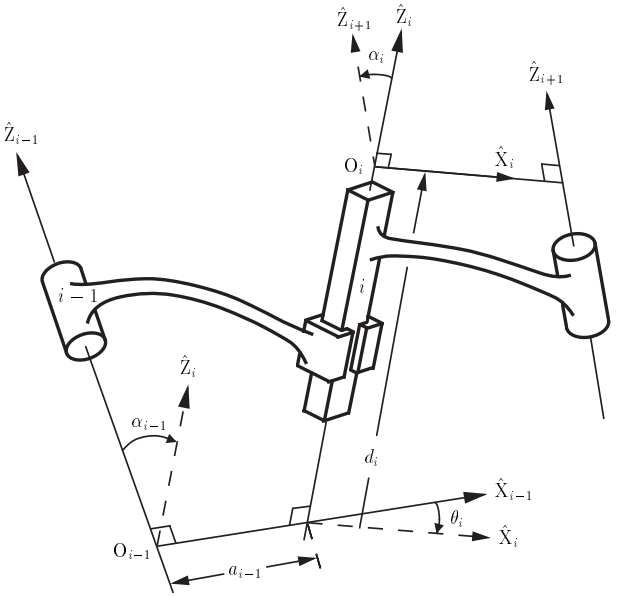

Figure 3 – Prismatic joint parameters

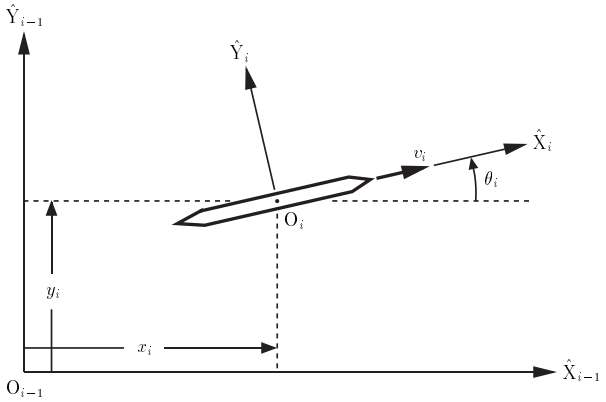

Figure 4 – Knife joint parameters

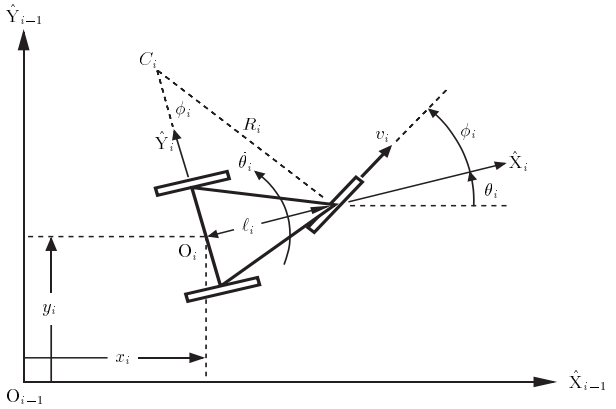

Figure 5 – Tricycle joint parameters

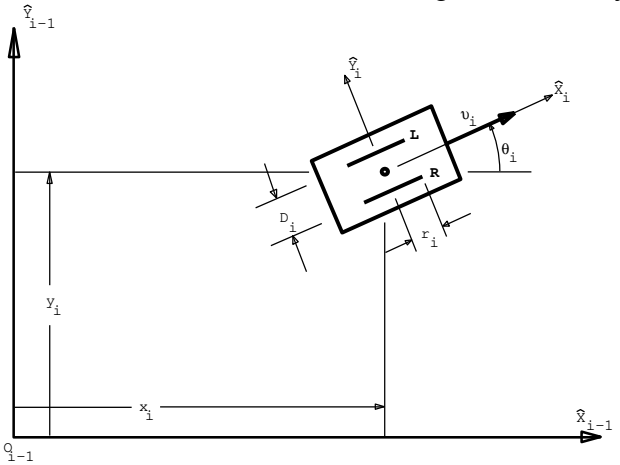

Figure 6 – Dual Wheel joint parameters

forward,backward, or rotational motion is allowed, but not sideways. (See Figure 4)

Tricycle Joint (T): x,y,z, theta (link), phi (wheel), length
Used for modeling forklifts, automobiles, and MoRT (See Figures 5and 8). For the turning wheel, an additional graphics description file "Wheel.gd" is needed. In Figure 8, "Wheel.gd" contains the desription of two thin horizontal cylinders (See GRAPHICS DESCRIPTION).

Dual Wheel Joint (D): x,y,z, theta, wheel_radius, axle separation
modeled after SWAMI: a mechanism driven by two parallel motorized wheels (See Figures 6 and 7).

==========

==========

After the joint specification and its parameters add the following:

graphics filename, sensor filename (if any), sensor numbers (if any),
mass, center_of_mass vector (x, y, and z values), inertia matrix (3x3)

If there are no sensors put "none" in place of sensorfilename and sensor numbers. The center of mass vector is from the joint's origin to its center of mass.

Example 1: Mechanism description – SWAMI: 24 sensors, estimated mass properties (Fig. 7)

SWAMI is modeled as a Dual Wheel joint with an initial location of (–850, –1850, 0) in the environment at an angle of 1.595 radians from the environment x–axis. Wheel radius and axle seperation are 152 mm and 406 mm, respectively. SWAMI's graphics description is found in the file "SWAMI.gd." SWAMI's sensor description filename is followed by a list of sensor identification numbers: one for every sensor described in SWAMI.sd. SWAMI's mass is estimated to be 100000 g. The center of gravity is located (from the midpoint between the wheel centers) 15 mm toward the front, 20 mm towards the right wheel, and 25 mm up. The inertia matrix [3] is given by a 3 x 3 matrix. Note: the mass properties shown below are only place holders and are not realistic.

```
SWAMI
D  –850 –1850 0 1.595 152 406
 SWAMI.gd
 SWAMI.sd 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
 100000  15 20 25

 10 1  1
 1  15 1
 1  1  20
```

Example 2: Mechanism description – MoRT (Fig. 8):
MoRT is modeled as a Tricycle joint (base) followed by a series of Revolute joints (Shilling arm). The midpoint of the rear axle (origin of Tricycle joint)starts at the origin. The body starts at zero radians with respect to the x–axis). The turning

wheel ("Wheel.gd") starts at zero degrees with respect to the body. The distance between the front and rear axle is 1108 mm. MoRT's graphics description is contained in "Mort.gd." The base has a mass of 100,000 g and a center of gravity at the midpoint of the rear axle.

The Shilling arm has a Revolute joint base located 128 mm back and 914 up from the Tricycle origin. It rotates in the horizontal plane. It is initialized pointing toward the mechanism rear (3.1415 rad from a vector pointing forward on the vehicle).

Note: the mass properties shown below are only place holders and are not realistic.

```
Mort
T  0 0 0 0 0 1108 Mort.gd
100000   0 0 0

10 0 0
0 10 0
0 0 10

R  0 –128 914 3.1416 ShillingWaist.gd
10 0 0 0

10 0 0
0 10 0
0 0 10

R  1.5708 0 0 .4 ShillingShoulder.gd
10 0 0 0

10 0 0
0 10 0
0 0 10

R  0 927 0 –.4 ShillingElbow.gd
10 0 0 0

10 0 0
0 10 0
0 0 10

R  0 508 0 0 ShillingWrist1.gd
10 0 0 0

10 0 0
0 10 0
0 0 10

R  –1.5708 0 0 0 ShillingWrist2.gd
10 0 0 0

10 0 0
0 10 0
0 0 10
```

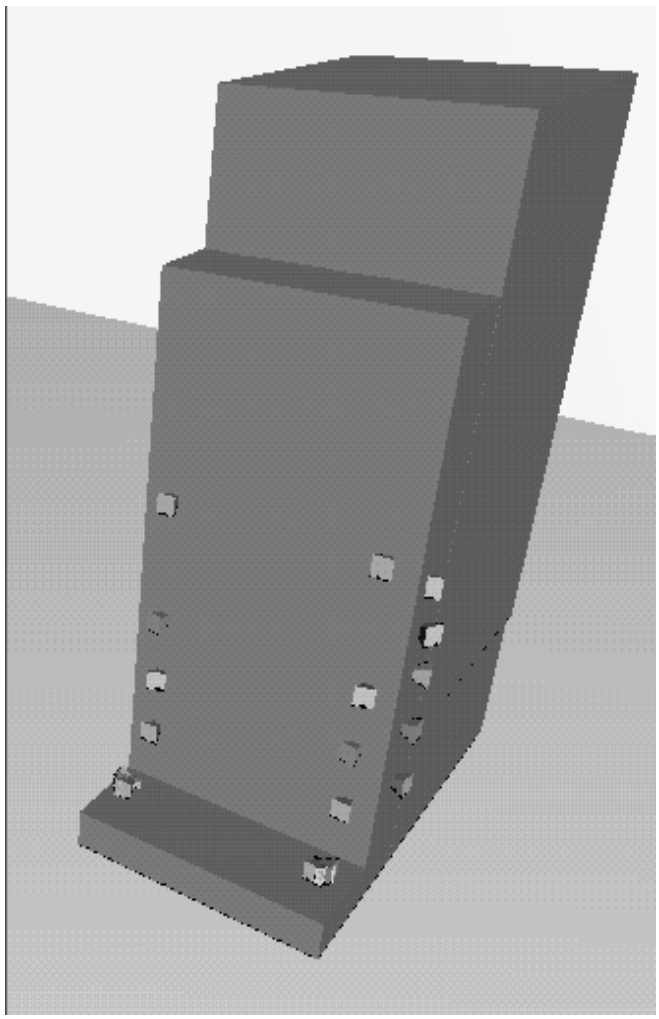

Figure 7– SWAMI

U 0 330 0 0 ShillingHand.gd
10 0 0 0

10 0 0
0 10 0
0 0 10

==========

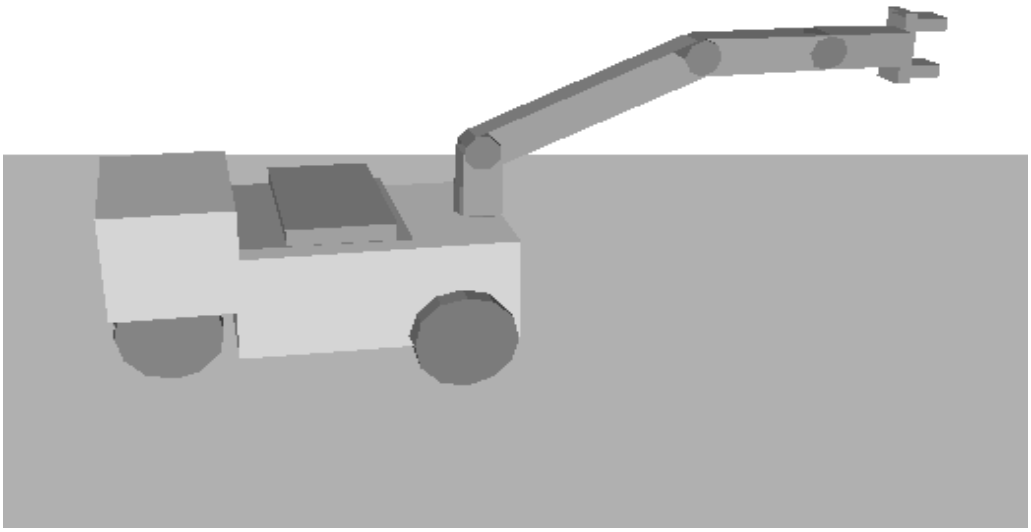

Figure 8 – MoRT

GRAPHICS DESCRIPTION

All mechanism dimensions (e.g., the Shilling arm graphics descriptions above) and environmental objects must be approximated by a suitable number (determined by the user) of graphics primatives or elements in a graphics description file. Currently available elements for this file are Plane, Rectangular Prism, Frustum, Cylinder, and Cone.

==========

Graphics_Elements and parameters

Plane(Pl): (origin at center)	x, y, color number, transform
Rectangular Prism(RP) (origin at center)	x, y, z, color number, transform
Cylinder(C): (origin at center, height along z−axis)	radius, height, number of side faces, color number, transform
Frustum(Fr): (origin at center, length along x−axis)	length, basewidth, base height,endwidth, endheight, color number, transform
Cone(C): (origin at center, height along z−axis)	one radius,height,number of faces,color number,transform

Color Table*

Color	Number
black	0
white	1
dark_grey	2
grey	3
light_grey	4
brown	5
light_blue	6
light_green	7
yellow	8
light_yellow	9
red	10

*Can be expanded, if needed

==========

Graphics transform consists of a rotation matrix and a position. The rotation is about the origin of the Graphics_Element. Taking three consecutive rotations about a fixed set of axes yields the general rotation matrix form [3]:

$$\begin{matrix} c\alpha c\beta & c\alpha s\beta s\gamma - s\alpha c\gamma & c\alpha s\beta c\gamma + s\alpha s\gamma \\ s\alpha c\beta & s\alpha s\beta s\gamma + c\alpha c\gamma & s\alpha s\beta c\gamma - c\alpha s\gamma \\ -s\gamma & c\beta s\gamma & c\beta c\gamma \end{matrix}$$

where $c\alpha$ represents the cos(α), $s\alpha$ represents sin(α), etc. The rotations must be about the X axis first (α), then Y (β), then Z (γ). This matrix simplifies greatly for the degenerate cases.

The position is a vector from the joint origin (depends on joint type) to the element center.

==========

Example: Transform – no rotation (identity matrix) and [1 2 3] position vector

```
1 0 0          <– Rotation matrix (identity)
0 1 0
0 0 1

1 2 3          <– Position vector (x y z) in mm
```

Example: Graphics Description file – SWAMI modeled with 4 Rectangular Prisms (See SWAMI Fig. 7)

```
RP  203 711 1372 6          <– Rectangular Prism of 203 by 711 by 1372 mm
                               color 6 (light_blue)
   1 0 0
   0 1 0                    <– Identity rotation matrix
   0 0 1

   483 0 686                <– Position vector from joint origin to RP center


RP  762 711 356 6

   1 0 0
   0 1 0
   0 0 1

   0 0 178                  <– Position vector from joint origin to RP center

RP  813 711 1346 6

   1 0 0
   0 1 0
   0 0 1

  –25 0 1029                <– Position vector from joint origin to RP center

RP  254 711 102 6

   1 0 0
   0 1 0
   0 0 1

   712 0 51                 <– Position vector from joint origin to RP center
```

==========

SENSOR DESCRIPTION

Sensor description files are used to attach sensors to links. A sensor graphic element is automatically placed on the mechanism at the specified position and orientation.

There needs to be a one−to−one correspondence between the sensors in this file and the sensor identification number list in the mechanism description file. This format was chosen so that a single sensor description file could be used for different links. Note that sensor placement does not depend upon the geometry of link. It is therefore possible to place sensors inside a mechanism or "off in space."

==========

Sensor Description Files (.sd)

Rangefinder: Rangefinder, maximum(1000)
rotation matrix (3x3)
position vector

Ultrasonic: Ultrasonic, maximum range, ultrasonic cone angle in radians
rotation matrix (3x3)
position vector

Example 1: Sensor Description file − SWAMI 24 sensor array numbered 1 to 24 consecutively, cone angle of .1 radians (See Fig. 9). All sensors are included for reference.

−−−−−−−−−−

```
Ultrasonic 1000 .1          <−Ultrasonic sensor with cone angle of .1 radians
                                  "Ultrasonic" must have a capital U.
  1 0 0                     <−Identity rotation matrix
  0 1 0
  0 0 1

  584 279 254               <−Position vector from link center to sensor center


Ultrasonic 1000 .1

  1 0 0
  0 1 0
  0 0 1

  584 −279 254



Ultrasonic 1000 .1

  .965926 .258819 0         <−Rotation matrix, −15 degrees about Z−axis
  −.258819 .965926 0              (0 degrees about X and Y)
  0 0 1
```

584 279 406

Ultrasonic 1000 .1

.965926 –.258819 0
.258819 .965926 0
0 0 1

584 –279 406

Ultrasonic 1000 .1

.965926 –.258819 0
.258819 .965926 0
0 0 1

584 279 559

Ultrasonic 1000 .1

.965926 .258819 0
–.258819 .965926 0
0 0 1

584 –279 559

Ultrasonic 1000 .1

1 0 0
0 1 0
0 0 1

584 279 864

Ultrasonic 1000 .1

1 0 0
0 1 0
0 0 1

584 –279 864

Ultrasonic 1000 .1

.130527 .991445 0
–.991445 .130527 0
0 0 1

483 –355 299

Ultrasonic 1000 .1

.130527 –.991445 0
.991445 .130527 0

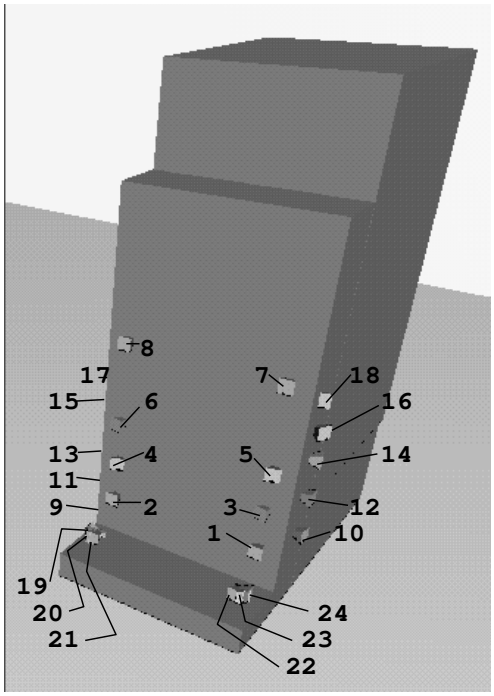

Figure 9– SWAMI sensor identification numbers.

<–Rotation matrix, –82.5 degrees about Z axis

0 0 1

483 355 299

Ultrasonic 1000 .1

0 1 0
–1 0 0
0 0 1

483 –355 444

Ultrasonic 1000 .1

0 –1 0
1 0 0
0 0 1

483 355 444

Ultrasonic 1000 .1

–.258818 .965926 0
–.965926 –.258818 0
0 0 1

483 –355 584

Ultrasonic 1000 .1

–.258818 –.965926 0
.965926 –.258818 0
0 0 1

483 355 584

Ultrasonic 1000 .1

.5 .866025 0
–.866025 .5 0
0 0 1

483 –355 692

Ultrasonic 1000 .1

.5 –.866025 0
.866025 .5 0

```
   0 0 1

   483 355 692

Ultrasonic 1000 .1

   .965926 .258819 0
   –.258819 .965926 0
   0 0 1

   483 –355 806

Ultrasonic 1000 .1

   .965926 –.258819 0
   .258819 .965926 0
   0 0 1

   483 355 806

 Ultrasonic 1000 .1

   0 1 0
  –1 0 0
   0 0 1

   711 –279 152

Ultrasonic 1000 .1

   .707107 .707106 0
   –.707106 .707107 0
   0 0 1

   711 –279 152

 Ultrasonic 1000 .1

   1 0 0
   0 1 0
   0 0 1

   711 –279 152

 Ultrasonic 1000 .1

   1 0 0
   0 1 0
   0 0 1

   711 279 152

 Ultrasonic 1000 .1
```

```
  .707107 –.707106 0
  .707106 .707107 0
  0 0 1

  711 279 152

Ultrasonic 1000  .1

  0 –1 0
  1 0 0
  0 0 1
  711 279 152
```

Example 2: Sensor Description file – Various sensor examples

```
Rangefinder 1000                    <–Rangefinder sensor, maximum range, capital
                                          R, the rest must be lower case

  .707107 –.707106 0                <–Rotation matrix, +45 degrees about Z axis
  .707106 .707107 0
  0 0 1

 10 20 30


Ultrasonic 1000 .212                      <–Ultrasonic sensor with maximum range of
                                                1000, cone angle of .212 radians

  0.852869    –0.150384    0.5         <–Rotation matrix, X rotation of 20 degrees
  0.331588    0.895721     –0.296198         Y rotation of 30 degrees
 –0.403317    0.418412     0.813798          Z rotation of 10 degrees

987 654 321
============================================================
```

MAP DESCRIPTION

The user has the ability to tailor sensor range data output as needed. Map dimensions, scale, initial value, and sensors can be adjusted. Map is currently displayed in a text window. Therefore, the map dimensions should reflect your output window size.

A small map with long sensor readings is impractical. Therefore adjust the scale to fit in most of the readings. The scale value is the side dimensions of a square cell.

The user specifies which sensors will update this map by listing their identification numbers (see MECHANISM DESCRIPTION). This is useful if it is desired to have maps for different heights (e.g., high, medium, low).

Some useful map commands are UpdateMap and ClearMap. UpdateMap() activates the sensors as given in the the map description file and adds the data to

the array. ClearMap(num) wipes the map clean and inserts the value "num" into each cell. This is currently necessary because Map has no memory (Map has memory now if users want to): it has no idea which way the vehicle is moving.

==========

Map Description Files (.mp)

map width, map height
ping on, range scale, initial fill value
mechanism name, sensor ID number list

Example: Map Description File – SWAMI.mp

68 50 *<–Map width of 68 numbers by 50 high*
1 100 1 *<–1 for ping on, Cell is .1 m by .1m. Initial cell value is 1*

SWAMI 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24

All 24 sensors on SWAMI will be read

==========

HISTOGRAM Information:

In addition to map, Histogram accumulates data in a display that does not move with the vehicle. The color of a cell in the histogram represents the cumulative sensor readings of a point fixed in space. It is very useful to help an operator to watch for possible collision based on continuous sensor reading. Usually, histogram will be only used with a map. To declare a histogram:

Histogram histo_name("Title", int xsize, int ysize, int Magnify);

See Main Steps For Coding and MBSim Application for more information on how to use it.

CONFIGURATION DESCRIPTION

This is an optional script file which defines the initial configuration (time = 0) of the mechanism. This is useful when starting a mechanism from a configuration different from that which was developed in the mechanism description file. In this way, the mechanism description file does not need to be redone with each new configuration.

==========

Configuration Files (.cfg) (joint values) (Use for initial conditions)

Revolute Joint (see Fig. 2): $\theta_{position}$, $\theta_{velocity}$,$\theta_{acceleration}$, joint torque

Prismatic Joint (see Fig. 3): $d_{position}$, $d_{velocity}$, $d_{acceleration}$, joint force

Unactuated Joint: (none)

Knife Joint (see Fig. 4):	x, y, $\theta_{position}$, body velocity, $\theta_{velocity}$, body acceleration, $\theta_{acceleration}$, force, torque
Tricycle Joint (see Fig. 5):	x, y, $\theta_{position}$,ϕ, body velocity, $\phi_{velocity}$, body acceleration, $\phi_{acceleration}$, force, torque
*Dual Wheel Joint (see Fig. 6):	x, y, $\theta_{position}$, $\phi_{velocity}$ (right wheel), $\phi_{velocity}$ (left wheel),$\phi_{acceleration}$ (right wheel), $\phi_{acceleration}$ (left wheel),torque (right wheel), torque (left wheel)

* ϕ for Dual Wheel is the angle of rotation of a motorized wheel.

Example: Configuration file – Tricycle with one revolute joint
Tricycle is initialized at no velocity or acceleration with the middle of its rear axis on the origin. Its main body axis lies along the x–axis. The steering wheel is straight.

```
0 0 0 0  0 0  0 0  0 0        <– Tricycle joint parameters
0  0  0  0                    <– Revolute joint parameters
==========================================================
```

VELOCITY DESCRIPTION

Motion of mechanism joints is currently handled by a velocity file. Work is underway on a controller that provides joint velocities. Velocities are linearly interpolated over the time interval so acceleration is constant.

```
==========================================================
```
Velocity Files (.vel)
Note: Each line contains all Mechanism velocity values with the first value on each line being time.

Revolute Joint (see Fig. 2):	$\theta_{velocity}$
Prismatic Joint (see Fig. 3):	$d_{velocity}$
Unactuated Joint:	(none)
Knife Joint (see Fig. 4):	body velocity, $\theta_{velocity}$
Tricycle Joint (see Fig. 5):	body velocity, $\phi_{velocity}$
Dual Wheel Joint (see Fig. 6):	$\phi_{velocity}$ (right wheel), $\phi_{velocity}$ (left wheel)

Example: Velocity file – MoRT with 5 DOF Shilling Arm

```
0.5 50   0     0   0   0   0   0
1.0 100   0   –.1  .5 –.5   0   0
1.5 200  0    –.3  .5 –.5   0   0
2.0 400 –.2   –.3  .1 –.5   0   0
2.5 400 –.3   –.3  .2  –.5   0   0
3.0 400 –.3   –.3  .2  –.5   0   0
3.5 400 –.3   –.3  .2  –.5  –.3   0
4.0 400 –.2    0   0   0  –.4   0
4.5 400 –.2   0   0   0  –.4   0
5.0 400 –.1   0   0   0   0  .2
5.5 400 –.1   0   0   0   0  .3
6.0 400 –.2   0   0   0   0  .3
6.5 200  0    0   0   0   0  .3
7.0 100  0   0   0   0   0  .3
7.5 100  0   0   0   0   0  .3
8.0 100  0   0   0   0   0  .3
8.5 100  0   0   0   0   0  .3
9.0 100  0   0   0   0   0  .3
9.5 100  0   0   0   0   0  .3
10.0 100  0   0   0   0   0  .3
 ^       ^    ^   ^   ^    ^   ^   ^
 |       |    |   |   |    |   |   |_ Outermost revolute link thetadot
 |       |    |   |   |    |   |
 |       |    |   |   |    |   |_____ Revolute link thetadot
 |       |    |   |   |    |
 |       |    |   |   |    |_________ Revolute link thetadot
 |       |    |   |   |
 |       |    |   |   |_______________ Revolute link thetadot
 |       |    |   |
 |       |    |   |___________________ Base revolute link thetadot
 |       |    |
 |       |    |_____________________________ Tricycle phidot (steering wheel turning rate)
 |       |
 |       |_____________________________________Tricycle joint velocity
 |
 |_____________________________________ Time at end of velocity
```

==========

POSITION DESCRIPTION

Animation of mechanisms can be performed by reading a script file which contains joint positions. Position script files are useful when animation is desired from data generated by an outside program. Animation of the outside data can give the user a better understanding of the results. The required joint values for each Link are shown below.

==========

Position Files (.pos)

Note: Each line contains all Mechanism position values with the first value on each line indicating the time.

Revolute Link (see Fig. 2): θ
Prismatic Link (see Fig. 3): d
Unactuated Link: (none)
Knife Link (see Fig. 4): x, y, θ
Tricycle Link (see Fig. 5): x, y, θ, ϕ
Dual Wheel Link (see Fig. 6): x, y, θ

Example 1: Position file – Knife

```
0.5  –1341.16  –1555.31  1.5
1.0  –1323.47  –1305.94  1.5
1.5  –1305.79  –1056.57  1.5
2.0  –1272.63  –809.267  1.375
2.5  –1194.41  –573.872  1.125
3.0  –1060.39  –365.146  0.875
 ^        ^         ^          ^
 |        |         |          |
 |        |         |          |__ Tricycle θ_position (body rotation)
 |        |         |
 |        |         |__ Tricycle y (y axis position)
 |        |
 |        |__ Tricycle x (x axis position)
 |
 |__ Time at this position
```

Example 2: Position file – MoRT (tricycle) with 5 DOF Shilling Arm (revolute joints)

```
0.5   12.5     0.00   0.000   0.00 3.14  0.400 –0.400  0.0  0.0
1.0   50.0     0.00   0.000   0.00 3.12  0.525 –0.525  0.0  0.0
1.5  125.0     0.00   0.000   0.00 3.02  0.775 –0.775  0.0  0.0
2.0  274.87 –0.45 –0.005 –0.05 2.87  0.925 –1.025  0.0  0.0
 ^      ^        ^       ^       ^     ^       ^       ^     ^     ^
 |      |        |       |       |     |       |       |     |     |
 |      |        |       |       |     |       |       |     |     |__ 5th revolute joint θ_pos.
 |      |        |       |       |     |       |       |     |
 |      |        |       |       |     |       |       |     |__ 4th revolute joint θ_position
 |      |        |       |       |     |       |       |
 |      |        |       |       |     |       |       |__ 3rd revolute joint θ_position
 |      |        |       |       |     |       |
 |      |        |       |       |     |       |__ 2nd revolute joint θ_position
 |      |        |       |       |     |
 |      |        |       |       |     |__ 1st revolute joint θ_position
 |      |        |       |       |
 |      |        |       |       |__ Tricycle φ (steering angle)
 |      |        |       |
 |      |        |       |__ Tricycle θ_position (body rotation)
 |      |        |
 |      |        |__ Tricycle y (y axis position)
 |      |
 |      |__ Tricycle x (x axis position)
 |
 |__ Time at this position
```

==========

ENVIRONMENT DESCRIPTION

"Enviroment" refers to the fixed surroundings. Building these description files is a process similar to that of mechanism description files where the environment is composed of inanimate objects. Each object has a graphics description file associated with it (See GRAPHICS DESCRIPTION above).

Environment Files (.env)

Object (O): x, y, z, graphics filename

Example: Environment file – Pallet.gd is drawn at the origin

O 0 0 0 Pallet.gd

Example: Environment graphics file: Pallet.gd – Pallet(Brown RP) with 4 Barrels(C of 11 sides) . Three are shown in Figure 10.

RP 1000 1000 150 2 *<–Rectangular Prism of 1m x 1m x .15m and color 2 (brown, see GRAPHICS DESCRIPTION)*

1 0 0 *<–Identity rotation matrix*
0 1 0
0 0 1

0 0 75 *<–offset center .075m (half of Pallet's thickness) up from the floor*

C 250 850 11 8

1 0 0
0 1 0
0 0 1

250 250 575

C 250 850 11 9

1 0 0
0 1 0
0 0 1

250 –250 575

C 250 850 11 6

1 0 0
0 1 0
0 0 1

–250 250 575

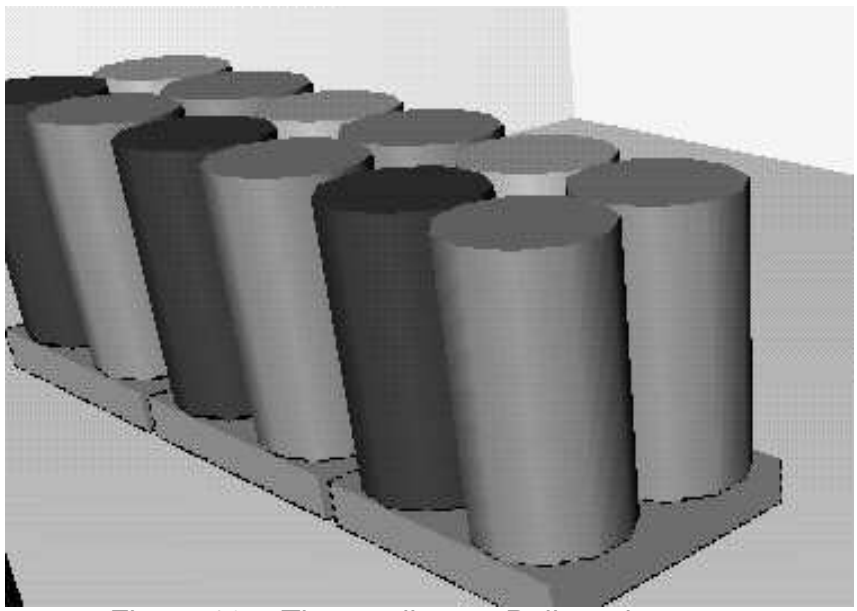

Figure 10 – Three adjacent Pallet.gd.

C 265 850 11 3

1 0 0
0 1 0
0 0 1

–250 –250 575

GRAPHICS VIEW

A view file defines the graphics window size, the "eye" position, and the point the eye is looking toward (point of attention). The graphics window size requires an x and y input: values between 500 and 1200 are best. "Eye" position is a three dimensional point in millimeters, as is the point of attention.

==========

View Files (.view)

window length, window height

x_{eye}, y_{eye}, z_{eye}
x_{poa}, y_{poa}, z_{poa}

Example: View File

700 700	*<–Square graphics window of 700 x 700*
–3500 1500 4000	*<–eye position at –3.5 m, 1.5 m, 4 m*
0 0 0	*<–point of attention, eye is looking toward the origin.*

==========

FORCE–TORQUE PLOTS

Do your joint velocities saturate the mechanism actuators? It is possible to plot torque/force plot from the joints in graphical form. Several steps must be followed: create plot object, read in plot limits and send data to plot. An example is studied step by step. This example is then implemented in a sample program.

==========

Example: Force/Torque Plot Tricycle,

In main()
Set up Plot object for each force/torque to be plotted. Give the plot a name in this declaration (e.g., "Torque").

```
Plot torqueplot("Torque");
```

Read in y–value force/torque plot limits.

```
   torqueplot.SetLimits("torquet.plot");
```

Sample torqueright.plot:

```
  –500 1500                    <– lower limit then upper limit
```

Declare a list of joint values. This list will hold the y plot value.

```
  Iterator<Joint_Value> forces_iter;  // for torque plots
```

Calculate current inverse dynamics of Tricycle and put in forces_iter

```
      forces_iter = tricycle.InverseDynamics(zeros, zeros);
```

Draw force and torque plots

```
      forces_iter.Head();     // Go to head of forces_iter
      torque = forces_iter()–>JointValue();
      forces_iter++;

      time = tricycle.Time();      // Time stamp tricycle torque data
      cout << "Time: " << time << "\n";
      torquetplot.Draw(time, torque);
```

==========

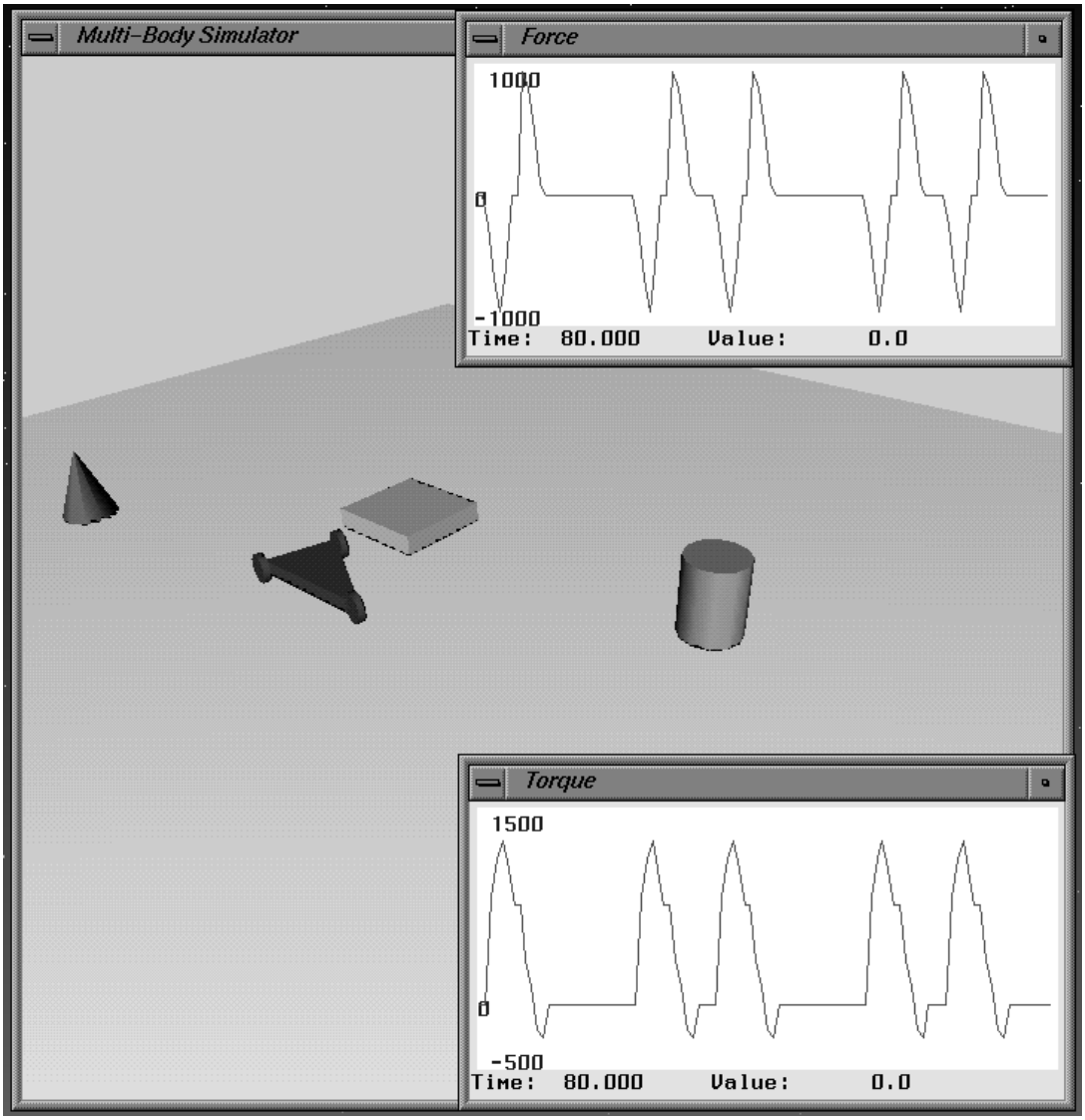

Figure 11 – Tricycle with Force (for base) and Torque (for front wheel) plots.