Hierarchical Transformations and Models
Overview

- Previously – Transformation
  - Translate, Rotate, Scale (shearing)

- Creating a scene: modeling and transformation

- Hierarchical modeling
  - Place model pieces relative to other model pieces
  - Place objects relative to other objects
  - Tree representation and traversal

- Examples
  - Robot arm
  - Humanoid robot
  - Scene graphs
Modeling

• A model has a default size, position, and orientation
• The modeling process determines the default shape of a model
• The modeling process is done by the application (it’s not unique)

Create a unit cylinder with its origin at (0,0,0)
Object Transformation

• Start with creating an object from a model
• The objects created from the same model has the same default shape
• But the object in the scene can become different later because of:
  – its own scale, rotate, and translate
Object Table

Store an object by assigning a number to each model and storing the parameters for its transformation

<table>
<thead>
<tr>
<th>Model</th>
<th>Scale</th>
<th>Rotate</th>
<th>Translate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$s_x$, $s_y$, $s_z$</td>
<td>$\theta_x$, $\theta_y$, $\theta_z$</td>
<td>$d_x$, $d_y$, $d_z$</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Scene example

A space = A coordinate system

multiple instances in world space (each has its own local space)

model/symbol in local space
Scene example

- 1 scale
- 1 rotate
- 1 translate

for each object

- In the local space, objects are identical. So we can use the same modeling process.
How to draw a robot (in a bad way...)

Images by Denis Zorin
How to draw a robot (in a bad way...)

Images by Denis Zorin
Draw a robot

• Positioning individual parts is hard
• If the whole robot moves
  – Reposition everything
• If only the upper arm moves
  – Reposition the upper and lower arm

• Divide the robot into pieces!
  – Combine as a hierarchy
Review: Tree

• Graph in which each node (except the root) has exactly one parent node
  – May have multiple children
  – Leaf or terminal node: no children
Robot as a hierarchy
a.k.a. tree or directed acyclic graph

Images by Denis Zorin
Modeling with Trees/Graphs

• Nodes
  – What to draw
  – Pointers to children

• Edges
  – Transformation from the parent node to a child node
Robot Arm

robot arm

parts in their own coordinate systems
Articulated Models

- Robot arm is an example of an articulated model
  - Parts connected at joints
  - Can specify state of model by giving all joint angles
Relationships in Robot Arm

- Base rotates independently
  - Single angle determines position
- Lower arm attached to base
  - Its position depends on rotation of base
  - Must also translate relative to base and rotate about connecting joint
Relationships in Robot Arm

• Upper arm attached to lower arm
  – Its position depends on both base and lower arm
  – Must **translate** relative to lower arm and **rotate** about joint connecting to lower arm
Required Matrices

• Rotation of base: \( \mathbf{R}_b \)
  – Apply \( \mathbf{M} = \mathbf{R}_b \) to base

• Translate lower arm relative to base: \( \mathbf{T}_{la} \)

• Rotate lower arm around joint: \( \mathbf{R}_{la} \)
  – Apply \( \mathbf{M} = \mathbf{R}_b \mathbf{T}_{la} \mathbf{R}_{la} \) to lower arm

• Translate upper arm relative to lower arm: \( \mathbf{T}_{ua} \)

• Rotate upper arm around joint: \( \mathbf{R}_{ua} \)
  – Apply \( \mathbf{M} = \mathbf{R}_b \mathbf{T}_{la} \mathbf{R}_{la} \mathbf{T}_{ua} \mathbf{R}_{ua} \) to upper arm
Tree Model of Robot

• It shows the relationships between parts of model
  – Can change “look” of parts easily without altering relationships

• Simple example of tree model

• Want a general node structure for nodes
A Chain of Transformations

\[
\begin{pmatrix}
x_e \\
y_e \\
1
\end{pmatrix} =
\begin{pmatrix}
T \\
0 \\
1
\end{pmatrix}
\]

\[
T = (rot\theta_2)(transl_1)(rot\theta_1)(transl_2)
\]

\[
= \begin{pmatrix}
\cos \theta_1 & -\sin \theta_1 & 0 \\
\sin \theta_1 & \cos \theta_1 & 0 \\
0 & 0 & 1
\end{pmatrix} \begin{pmatrix}
1 & 0 & l_1 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix} \begin{pmatrix}
\cos \theta_2 & -\sin \theta_2 & 0 \\
\sin \theta_2 & \cos \theta_2 & 0 \\
0 & 0 & 1
\end{pmatrix} \begin{pmatrix}
1 & 0 & l_2 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\]
Thinking of Transformations

- As changing the world space into the local space

\[ T = (\text{rot}_{\theta_1})(\text{transl}_{l_1})(\text{rot}_{\theta_2})(\text{transl}_{l_2}) \]

\[
= \begin{pmatrix} \cos \theta_1 & -\sin \theta_1 & 0 \\ \sin \theta_1 & \cos \theta_1 & 0 \\ 0 & 0 & 1 \end{pmatrix}
\begin{pmatrix} 1 & 0 & l_1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}
\begin{pmatrix} \cos \theta_2 & -\sin \theta_2 & 0 \\ \sin \theta_2 & \cos \theta_2 & 0 \\ 0 & 0 & 1 \end{pmatrix}
\begin{pmatrix} 1 & 0 & l_2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}
\]
Thinking of Transformations

• As changing the world space into the local space

\[
T = \begin{pmatrix}
\cos \theta_1 & -\sin \theta_1 & 0 \\
\sin \theta_1 & \cos \theta_1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
1 & 0 & l_1 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
\cos \theta_2 & -\sin \theta_2 & 0 \\
\sin \theta_2 & \cos \theta_2 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
1 & 0 & l_2 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\]
Thinking of Transformations

• As changing the world space into the local space

\[
T = (\text{rot}_{\theta_1})(\text{transl}_{l_1})(\text{rot}_{\theta_2})(\text{transl}_{l_2})
\]

\[
= \begin{pmatrix}
\cos \theta_1 & -\sin \theta_1 & 0 \\
\sin \theta_1 & \cos \theta_1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
1 & 0 & l_1 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
\cos \theta_2 & -\sin \theta_2 & 0 \\
\sin \theta_2 & \cos \theta_2 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
1 & 0 & l_2 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\]
Thinking of Transformations

- As changing the world space into the local space

\[
T = (\text{rot}_{\theta_1}) (\text{trans}_{l_1}) (\text{rot}_{\theta_2}) (\text{trans}_{l_2})
\]

\[
\begin{pmatrix}
\cos \theta_1 & -\sin \theta_1 & 0 \\
\sin \theta_1 & \cos \theta_1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
1 & 0 & l_1 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
\cos \theta_2 & -\sin \theta_2 & 0 \\
\sin \theta_2 & \cos \theta_2 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
1 & 0 & l_2 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\]
Thinking of Transformations

• As changing the world space into the local space

\[
T = (rot_{\theta_1})(transl_1)(rot_{\theta_2})(transl_2)
\]

\[
= \begin{pmatrix}
\cos \theta_1 & -\sin \theta_1 & 0 \\
\sin \theta_1 & \cos \theta_1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
1 & 0 & l_1 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
\cos \theta_2 & -\sin \theta_2 & 0 \\
\sin \theta_2 & \cos \theta_2 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
1 & 0 & l_2 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\]
Thinking of Transformations

- In a fixed global space

\[
T = (\text{rot}\theta_1)(\text{transl}_1)(\text{rot}\theta_2)(\text{transl}_2) \\
\begin{pmatrix}
\cos \theta_1 & -\sin \theta_1 & 0 \\
\sin \theta_1 & \cos \theta_1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
1 & 0 & l_1 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
\cos \theta_2 & -\sin \theta_2 & 0 \\
\sin \theta_2 & \cos \theta_2 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
1 & 0 & l_2 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\]
Thinking of Transformations

• In a fixed global space

\[
T = (\text{rot}\theta_1)(\text{transl}_1)(\text{rot}\theta_2)(\text{transl}_2)
\]

\[
= \begin{pmatrix}
\cos \theta_1 & -\sin \theta_1 & 0 \\
\sin \theta_1 & \cos \theta_1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
1 & 0 & l_1 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
\cos \theta_2 & -\sin \theta_2 & 0 \\
\sin \theta_2 & \cos \theta_2 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
1 & 0 & l_2 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\]
Thinking of Transformations

• In a fixed global space

\[
T = (\text{rot}_{\theta_1})(\text{transl}_1)(\text{rot}_{\theta_2})(\text{transl}_2)
\]

\[
= \begin{pmatrix}
\cos \theta_1 & -\sin \theta_1 & 0 \\
\sin \theta_1 & \cos \theta_1 & 0 \\
0 & 0 & 1 \\
\end{pmatrix}
\begin{pmatrix}
1 & 0 & l_1 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
\end{pmatrix}
\begin{pmatrix}
\cos \theta_2 & -\sin \theta_2 & 0 \\
\sin \theta_2 & \cos \theta_2 & 0 \\
0 & 0 & 1 \\
\end{pmatrix}
\begin{pmatrix}
1 & 0 & l_2 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
\end{pmatrix}
\]
Thinking of Transformations

• In a fixed global space

\[
T = (\text{rot}_{\theta_1})(\text{transl}_{l_1})(\text{rot}_{\theta_2})(\text{transl}_{l_2})
\]

\[
\begin{pmatrix}
\cos \theta_1 & -\sin \theta_1 & 0 \\
\sin \theta_1 & \cos \theta_1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
1 & 0 & l_1 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
\cos \theta_2 & -\sin \theta_2 & 0 \\
\sin \theta_2 & \cos \theta_2 & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
1 & 0 & l_2 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\]
Thinking of Transformations

• In a fixed global space

\[ T = (\text{rot}_\theta_1)(\text{transl}_1)(\text{rot}_\theta_2)(\text{transl}_2) \]

\[ = \begin{pmatrix} \cos \theta_1 & -\sin \theta_1 & 0 \\ \sin \theta_1 & \cos \theta_1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & l_1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos \theta_2 & -\sin \theta_2 & 0 \\ \sin \theta_2 & \cos \theta_2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & l_2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \]
Robot as a hierarchy
(save intermediate transformations)

Images by Denis Zorin
The advantages of the hierarchy

- Transform multiple objects together!
- Duplicate multiple objects easily!
- See a demo
Scene Graph

• Convenient Data structure for scene representation
  – Transformations
  – Materials, color
  – Multiple instances
• Basic idea: Hierarchical Tree
• Useful for manipulation/animation
  – Especially for articulated figures
• Useful for rendering too
  – Multi-pass rendering, occlusion culling
Sample Scene
Hierarchical scene – chair 1
Hierarchical scene – chair 2
Hierarchical scene - chessboard
White chess pieces – child of chessboard
Rook piece – child of white chess pieces