The following are notes taken while watching a demo given by Takasi Yamamiya on his "Skeleton" etoy based constraint system.

October 9, 2003

Takasi Yamamiya
"Skeleton" -- First test with constraint system
A "play by play" on how he made his demos

Making the Skull
This is normal text. He takes regular text out of the supplies bin. He makes the text larger. He changes the point size to 36, then, using the green duplicate handle, he makes three or four copies and puts each text morph under the other. He now has three pieces of text in a vertical line.

Modifying Ned's "connectors" he takes lines and connects the lines into the rotation center of each piece of text. This is the basic form. Now they act like springs.
(The default line length is 100.)

Yamamiya created a "goals" flap that contains a "hardness" slider. Another thing you can change is the length of the line via the yellow handle.

Now he paints a skull. He attributes his "watercolor" style to his drawing teacher. He keeps the painting/skull and makes it smaller. He changes the center of rotation to the mouth area, and then uses "connect to relative point" to the top of the skull, so that when he rotates the skull, it "cranks" the connector lines so they can now act as springs.

He changes the heading increasing it by 10, and sets the clock ticking. Line "hardness" can be adjusted. Lines can be made shorter by adjusting with the yellow handle.

Difference between regular line and "weak" line.
Skeleton kicking soccer ball
Set hardness slider to zero to enable changes
Select a weak line. A "normal line" remembers its own length. You can't calculate the length. A weak line you can't set the length, but you can measure it. So in this script, we subtract the line length by 400 then the normal line moves.
(Line – on ball is "normal" – line used to manipulate is "weak".

Using connectors – choose “attachment point adjuster” He puts a star inside the adjuster and deletes all but one “adjustment point”. He removes star from the adjuster and attaches the weak line to the star at the adjustment point. He uses the star to manipulate the skeleton. Using the “be locked” menu item he locked the new star. He then grabs a “stars heading” tile and increases it by 5. Now the star rotates.
When you attach a line to another sketch
If you have a dot of paint using the “center” as the attachment point is better, for irregular shapes it’s best to use the point you want – pick any point on the irregular shape. If you click the end of the weak line, you receive the menu – go to “end” on that menu and then “connect to relative point”.

The Ellipse
The top script is straightforward. Keeping the sum of the two lengths the same.
Script 2 “fudge factor” uses “setRotationCenterFrom” 390@310 He rotates the point, and resets the center of rotation. The center of rotation (blue dot) should beset more or less to the center of the ellipse. The 390@310 script resets it each time. The blue dot is not creating the lines, but is part of the connector. He sets the pen down to true, but it writes a center point. This is why he chose to have the end of the connector to draw the point.

Script 1 was created to ‘smooth’ the movement of the dot along the ellipse. You have to look at the line end constraint in order to see where the drawing is being done. He added the feature of using translucency to extend the 8 bit color range to 32 bit color. High translucency hides the “jaggies”. Hasn’t found other uses for this yet.

Software as Art
To make a “fuzzball” – take a point, attach a normal line, make it’s heading random. Set it forward by “n”. Go to pen use, pen color, pen size, select translucent, speed it up to 25 ticks per second and let it go – wait until it makes a “good one”.

Midpoint lines
Make a Constraint
The midpoint is the average of the two coordinates. (Normal Etoy). He couldn’t synchronize using the regular Etoy system so he designed a new status. It’s a new event that tells everyone to compute. By using the brown “move” handle he changes the position of the morph – this causes the update. Instead of updating right away, it updates at the next step. By defining the update at the “next step” you can avoid infinite loops.

Creating a quadralateral with midpoints

Duplicate points by using siblings and make multiples. By changing point headings they all change. He wants to be able to control all the midpoints with one script and so he's used the make siblings feature. When you've finished making the copies, go to the viewer and see the scripts and variables – red and blue. Clicking on a point gives indicators. He goes to each midpoint and hooks up a red dot and a blue dot. Now they all "know each other". Use the brown handle to move the dots. Now he has enough dots and they are all connected. Now he adds "weak lines". He duplicates many weak lines and connects them to the dots. Sequence is of no importance. Lines can be locked to facilitate.

Bidirectional constraint
By clicking on the brown "move" handle, you can move the points.
This will allow to move midpoints to "solve" outside.
Select one dot, get its viewer. Go to basic, select "player" tile. (Yamamiya created). Set blue's player – red's player.

Geometry Theorem – a simple example with three colored dots
Blue, green, red
Green is called "point".
He has made a textual script designating other points in the instance variable field point's red and point's blue (as variables)
Red point has variables - Red's point and red's blue
Blue point also has similar variables so each dot has a reference to the other two
Mouse down for a few seconds for cross hair to disappear.
Yamamiya has created new events called "changed" and "updating".
Set each of the script's clocks to "changed".
Select the brown move handle to move all the dots.