Non-linear Perspective Widgets for Creating Multiple-View Images

Nisha Sudarsanam (Mindjet Corp)
Cindy Grimm (Washington Univ. in St. Louis)
Karan Singh (University of Toronto)
What is non-linear perspective?

Picasso’s Guitar, Glass and Fruit Dish

Composition of linear perspective views

Initial Scene

View_1

View_2

View_3

Final Scene
Motivation

• Linear perspective is sometimes restrictive
Combine to make one view

Unwrap widget

Fish-eye

Unwrap plus a fish-eye
User interface problem

• What do you want to see where?

Exploring a human pelvis
Our contribution

• Task breakdown for user
  – 3D selection of model subset
  – Desired viewing attributes
  – 2D placement in image

• Underlying framework for widget creation
  – Specify region of interest and fall-off
  – Support specific viewing effects
  – Automatic 2D placement algorithms
Previous work

• View transform as space deformation
  – “Bend” view rays in space
  – Modify perspective view to curve space
Previous work

• Manual placement of cameras
  – Blend to get camera for each point in space
    • Karan Singh, “A fresh perspective”, GI 2002
    • P. Coleman, K. Singh, L. Barrett, N. Sudarsanam, C. Grimm, “3D screen-space widgets for non-linear projection”, Graphite 2005
  – Our rendering approach
  – Very flexible, user-intensive
Key differences

• Local deformation specified wrt global camera
  – Can change global camera
Key differences

• Use interactive widgets
  – Specifies a specific view transformation
  – Encapsulates input parameters
  – Can string widgets together
Unwrap widget

• 3D volume, view direction
  – Look down vector
  – Placed on side of screen (default)
Unwrap widget

• Keeps view direction as camera changes
  – May move on screen
Fisheye widget

• Magnify selected region

m = 1.19
m = 1.45
m = 1.6
Clip widget

- Select area
  - Pulls to side

Destination area

Rotating the camera
Clip widget

Revealing the Knot
Panorama widget

- Input is key frames
Chaining and multiple widgets
Framework

- 3D region of influence
  - Fall-off
- Changes to the camera
- Destination area
  - Relative to bbox
  - COP
- Destination size
  - Bbox of region

- E.g., box (clip)
- E.g., Zoom (fisheye)
- E.g., To side (unwrap)
- Same or smaller (most)
System help

• Automatic routines for determining best destination location, size
  – Implemented as a Center of Projection change
  – Based on projection of bounding boxes

• Automatic weight fall offs
  – Multiple widgets: average effect based on weights
Rendering

- GPU implementation
- Calculate camera for each vertex
  - Based on 3D influences, local camera changes
  - Average projection from each camera
- Use original camera for lighting
Conclusion

- Encapsulate specific changes in widgets
- Framework for making and combining the widgets
- Easy to use, but not as flexible