Supplemental Material: Layout Design for Augmented Reality Applications

1 Script Examples

Following are two sample scripts from the FLARE framework. First a very simple script defining two objects X_1, X_2, where X_1 is on the floor and looking at X_2 which is above him. Finally there is line of sight between the objects:

```
// Object's Local Frame:
// Position - Origin
//  X axis   - Forward
//  Y axis   - Right
//  Z axis   - Up

// Height Difference
assert(
 Abs(LocalZ( X1.Frame, X2.Frame.Position ))
    > 0.3 );

// Object on the Floor
assert(
 LocalZ( Floor.Frame, X1.Frame.Position )
    = 0 );


// X1 is facing in the direction of X2,
//        regardless of height differences

assert(
 Between(LocalX(X1.Frame, X2.Frame.Position),
     2, 3));
assert(
 Between(LocalY(X1.Frame, X2.Frame.Position),
    -0.01, 0.01));

// Line of sight between the anchor point
//           of objects X1 and X2
assert( Visible( X1.Frame.Position,
   X1.Frame.Position ) = 0 );
```

The second script is the one we use for the Laplacian cycle quantitative experiment. Ten objects arranged on a horizontal surface such that there is a minimal distance of 3 between each object and the next, and the position of X_i approximates $(X_{i-1} + X_{i+1})/2$

```
assert( horizontal(X1) );
assert( horizontal(X2) );
assert( horizontal(X3) );
assert( horizontal(X4) );
assert( horizontal(X5) );
assert( horizontal(X6) );
assert( horizontal(X7) );
assert( horizontal(X8) );
assert( horizontal(X9) );
assert( horizontal(X0) );

assert( (Distance( X1.Frame.Position,
 X2.Frame.Position )) > 3 );
assert( (Distance( X2.Frame.Position,
 X3.Frame.Position )) > 3 );
assert( (Distance( X3.Frame.Position,
 X4.Frame.Position )) > 3 );
assert( (Distance( X4.Frame.Position,
 X5.Frame.Position )) > 3 );
assert( (Distance( X5.Frame.Position,
 X6.Frame.Position )) > 3 );
assert( (Distance( X6.Frame.Position,
 X7.Frame.Position )) > 3 );
assert( (Distance( X7.Frame.Position,
 X8.Frame.Position )) > 3 );
assert( (Distance( X8.Frame.Position,
X9.Frame.Position )) > 3 );
assert( (Distance( X9.Frame.Position,
 X10.Frame.Position )) > 3 );
assert( (Distance( X10.Frame.Position,
 X1.Frame.Position )) > 3 );

assert( Distance(
   (X1.Frame.Position+X3.Frame.Position) * 0.5,
 X2.Frame.Position ) = 0 );
assert( Distance(
   (X2.Frame.Position+X4.Frame.Position) * 0.5,
    X3.Frame.Position ) = 0 );
assert( Distance(
   (X3.Frame.Position+X5.Frame.Position) * 0.5,
    X4.Frame.Position ) = 0 );
assert( Distance(
   (X4.Frame.Position+X6.Frame.Position) * 0.5,
    X5.Frame.Position ) = 0 );
assert( Distance(
   (X5.Frame.Position+X7.Frame.Position) * 0.5,
    X6.Frame.Position ) = 0 );
assert( Distance(
   (X6.Frame.Position+X8.Frame.Position) * 0.5,
    X7.Frame.Position ) = 0 );
assert( Distance(
   (X7.Frame.Position+X9.Frame.Position) * 0.5,
    X8.Frame.Position ) = 0 );
assert( Distance(
   (X8.Frame.Position+X10.Frame.Position) * 0.5,
    X9.Frame.Position ) = 0 );
assert( Distance(
   (X9.Frame.Position+X1.Frame.Position) * 0.5,
    X10.Frame.Position ) = 0 );
assert( Distance(
   (X10.Frame.Position+X2.Frame.Position) * 0.5,
    X1.Frame.Position ) = 0 );
```

2 Photo Overlay

In order to demonstrate the flexibility of our framework, we adapt it to 2D layout design. When reviewing a large collection of photographs, it is often useful to see the corresponding meta-data (title, date, exposure information, geo-tags and user supplied tags) alongside the original image. In smaller form-factors, there is not enough screen real estate to display the image alongside the information. We present an application to overlay textual and visual information on an image, based on geometrical and aesthetic design rules.

In this application, the objects in the design are the different textual and visual elements to be superimposed, and the environment is the background image. Given an image, we extract textual meta-data from its EXIF such as title, camera and lens model, aperture, shutter speed as well as calculate a luminance histogram. We then calculate a saliency map [Perazzi et al. 2012], run an edge detection filter and extract the color palette of the image [Morse et al. 2007] to which

we add black and white (Figure 1.

The rules we employ for this design are such that the superimposed elements are positioned on the non-salient regions in the image and their colors are taken from the image extended color palette. Additionally, the title is larger than the exposure information, and the exposure information elements are ordered vertically and attempt to align (left or right depending on their position within the image). The results of our design can be seen in figure 2 and figure 3.

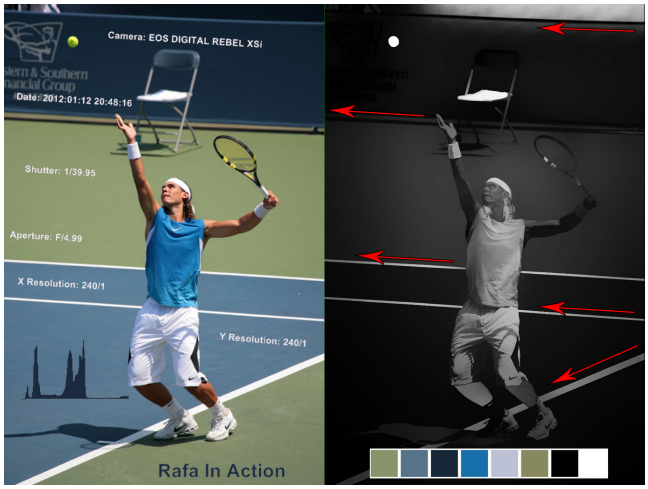

Figure 3: *In this example, in addition to the described design rules, we align the meta-data elements along the most prominent horizontal edges, extracted using edge-detection.*

References

MORSE, B., THORNTON, D., XIA, Q., AND UIBEL, J. 2007. Image-based color schemes. In *ICIP 2007*.

PERAZZI, F., KRÄHENBÜHL, P., PRITCH, Y., AND HORNUNG, A. 2012. Saliency filters: Contrast based filtering for salient region detection. In *CVPR 2012*.

Figure 1: *For each image we extract a saliency map [Perazzi et al. 2012] and a color palette [Morse et al. 2007].*

Figure 2: *Applying our design rules to various images produces pleasing results automatically and in real time, allowing a user to bring up information about an image without interrupting the flow of images.*