August 20, 2019

Anchoring

Anonymous Anonymous

Anchoring

Abstract

A system for recovering information, which is lost in a first transformation from a three-dimensional (3D) environment to a two-dimensional (2D) environment is disclosed. The information needs to be recovered because a second transformation is to be carried out from the 2D environment back to the 3D environment. The system recovers the information using anchors, which hold anchor nodes, while the second transformation is being carried out. The system includes a 2D view hierarchy having views and a 3D node hierarchy having 3D entities (i.e. 3D nodes and the anchor nodes). The 2D view hierarchy is an inverted tree structure with a root view positioned at top of it. The 2D view hierarchy and the 3D node hierarchy are associated with each other on a 3D panel. The 3D panel includes the 3D entities and the views. In the 3D panel, each of the views owns the anchor, which in turn owns the 3D node. The anchor is communicatively coupled to an anchor manager. The anchor manager holds a reference to the root view of the 2D view hierarchy of the 3D panel. The anchor manager determines a 2D transformation in position, scale and rotation from each of the views to the root view. A 2D point in each of the views in the 2D view hierarchy is matched to a 3D point at a same location on each of the 3D nodes. The 2D transformation is then used to carry out a required 3D transformation on the 3D point. In this mapping from the 2D view hierarchy to the 3D node hierarchy, the anchor manager controls a relationship between each of the views and the anchor. The anchor notifies the anchor manager when the relationship between each of the views and the anchor is established or broken. Consequently, the anchor manager adds or removes the anchor node as a sub-node in the 3D node hierarchy. This way, by means of the anchor manager, the 3D entities have positions relative to their 3D parentage in a 3D coordinate space, which are consistent with positions relative to their full logical parentage, which is inclusive of the views. Due to this, the system does not lose the information in positioning the 3D entities.

Problem statement

A user interface is usually created and presented in the 2D environment or a two-and-a-half-dimensional (2.5D) environment. For presenting content generated in the 3D environment, there is a need for its transformation into the 2D environment or the 2.5D environment (as per the user interface). As such, it is not always efficient to spend an app's complexity budget (costs that depend on complexity of an app) on managing minute details of (and bugs arising from) custom 3D transformations, because it might be difficult to deal with the custom 3D transformations. So, it is preferable to make use of conventional 2D layout algorithms and then transform back into the 3D environment from the 2D environment. For example, a UICore provides a panel class to bridge from the 3D environment into the 2D environment that hosts the views. The UICore is a 2D framework. Unfortunately, this transition to a lower-dimensional coordinate space results in a loss of the information within the 2D view hierarchy. Further, there is a loss of dimensionality in positioning, scaling and rotation.

In a previous art, each of the views in the 2D view hierarchy owns the 3D node in the 3D node hierarchy, thus providing a transition path for a transformation from the 2D environment to the 3D environment. The 3D node hierarchy is a tree hierarchy having the 3D nodes that represents the 3D environment. While transitioning from the 3D environment to the 2D environment, there is a loss of a coordinate as shown in Figure 1 below. Also, even if each of the views removes the 3D node owned by it, the 3D node remains in the 3D node hierarchy because it is also owned by another 3D node as its sub-node. Hence, the 3D node becomes an unwanted node. Further, the 3D node has a position relative to its 3D parentage in the 3D coordinate space that is inconsistent with a position relative to its full logical parentage, which is inclusive of the views. Due to this, the system loses the information in positioning the 3D node. This situation is referred to as a node leak as shown in the Figure 1 and this occurs due to a double ownership of the 3D node.

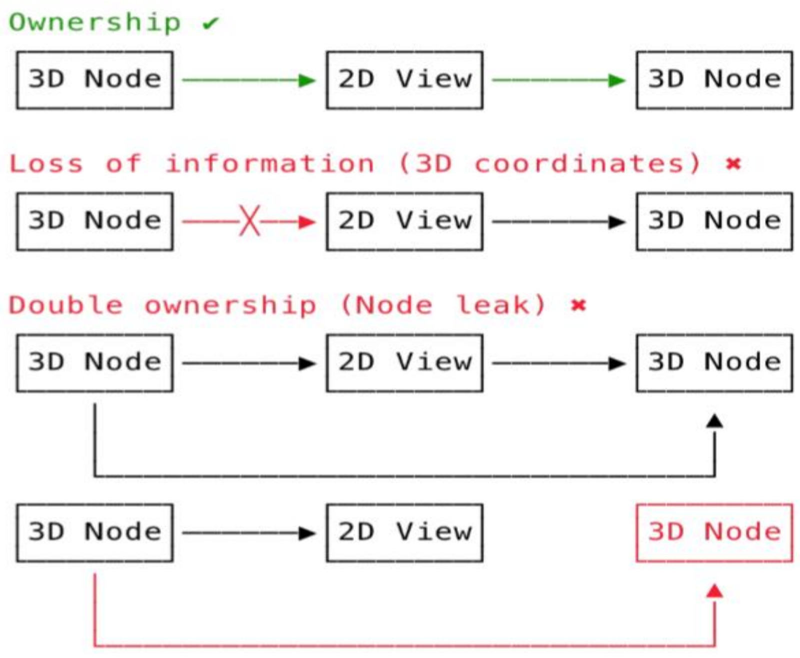

Figure 1: Challenges in 3D->2D->3D transition in the previous art

The present disclosure proposes a novel solution to overcome the above-mentioned challenges.

System and Working

The present disclosure describes a system to recover information, which is lost during a first transformation from a three-dimensional (3D) environment to a two-dimensional (2D) environment on a 2D user interface. The information needs to be recovered because a second transformation is to be carried out from the 2D environment back to the 3D environment. The system recovers the information using anchors that hold anchor nodes, while the second transformation is being carried out.

The system includes following entities:

- A 2D view hierarchy
- A 3D node hierarchy

The 3D node hierarchy is a tree hierarchy having 3D nodes that represents the 3D environment. Each of the 3D nodes includes sub-nodes, whereas the 3D nodes and the sub-nodes are structurally specified by a parent-child relationship. The 2D user interface is comprised of the 2D view hierarchy structured as an inverted tree having views, which further includes a root view positioned at top and sub-views positioned on branches below. The views and the sub-views are structurally specified by the parent-child relationship. The 2D view hierarchy and the 3D node hierarchy are associated with each other on a 3D panel. The 3D panel includes the views from the 2D view hierarchy and 3D entities (i.e. the 3D nodes and the anchor nodes) from the 3D node hierarchy as shown in Figure 2. The 3D panel is defined as an object that exists within a 3D coordinate space to present a 2D surface.

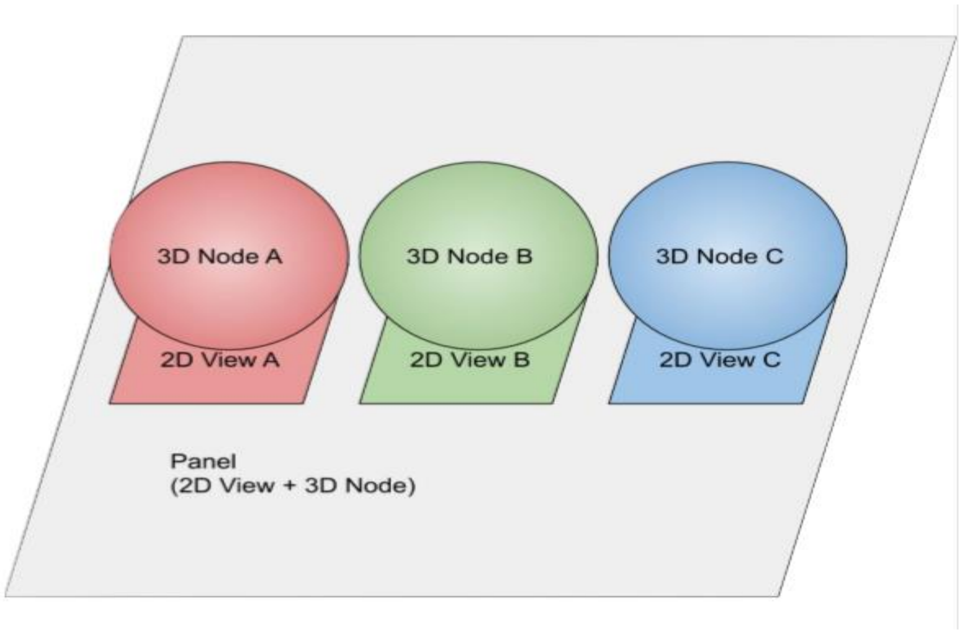

Figure 2: Architecture of the 3D panel

In the 2D view hierarchy, the root view owns the sub-views. Each view in the 2D user interface represents a rectangular area of a display. The views are responsible for everything that is drawn in the rectangular area and for responding to events that occur within the rectangular area (such as a touch event). The sub-views of each of the views are constrained to appear within bounds of the rectangular area of each of the views.

For the second transformation from the 2D environment to the 3D environment, each of the views is to be mapped to the 3D nodes in the 3D panel utilizing an anchor manager. The anchor manager is a piece of machinery that controls this mapping. There is one anchor manager for each 3D panel. The anchor manager holds a reference to the root view of the 2D view hierarchy of the 3D panel. The anchor manager determines a 2D transformation in position, scale and rotation from each of the views to the root view. The anchor manager then matches a 2D point on each of the views with a 3D point at a same location on each of the 3D nodes. Thereafter, the anchor manager carries out a corresponding 3D transformation on the 3D point in accordance with the 2D transformation. In order to match the 2D point with the 3D point, a linkage from the 2D view hierarchy to the 3D node hierarchy is provided by the anchor. The anchors are owned by the views and each anchor holds an anchor node. Anchor is a representation of a relationship between each view and the 3D node, whereas the anchor node is an actual entity that is inserted into the 3D node hierarchy. Each of the anchors creates the relationship from each view to the 3D node. This relationship prohibits the 3D nodes in the 3D node hierarchy from adding the anchor nodes as their sub-nodes. Only the anchor manager can add or remove the anchor nodes as the sub-nodes. This makes a relationship between the 3D node of the 3D panel and the anchor node weak, and this relationship is governed by a relationship between each of the views and the anchor owned by it. When the relationship between each of the views and the anchor is broken, the anchor notifies the anchor manager. Consequently, the anchor manager breaks the relationship between the 3D node of the 3D panel and the anchor node. Similarly, when the anchor is attached to any of the views, the anchor notifies the anchor manager. Consequently, the anchor manager adds the anchor node as the sub-node of the 3D node in the 3D node hierarchy as shown in Figure 3. Also, when a relationship between any of the views and the 3D panel is modified, i.e. due to the 2D transformation, the relationship between the 3D node of the 3D panel and the anchor node is also modified. This way, the 3D entities have a single parentage in the 3D node hierarchy and the 2D view hierarchy. The 3D entities have positions relative to their 3D parentage in the 3D coordinate space, which are consistent with positions relative to their full logical parentage, which is inclusive of the views. Due to this, the system does not lose the information in positioning the 3D entities.

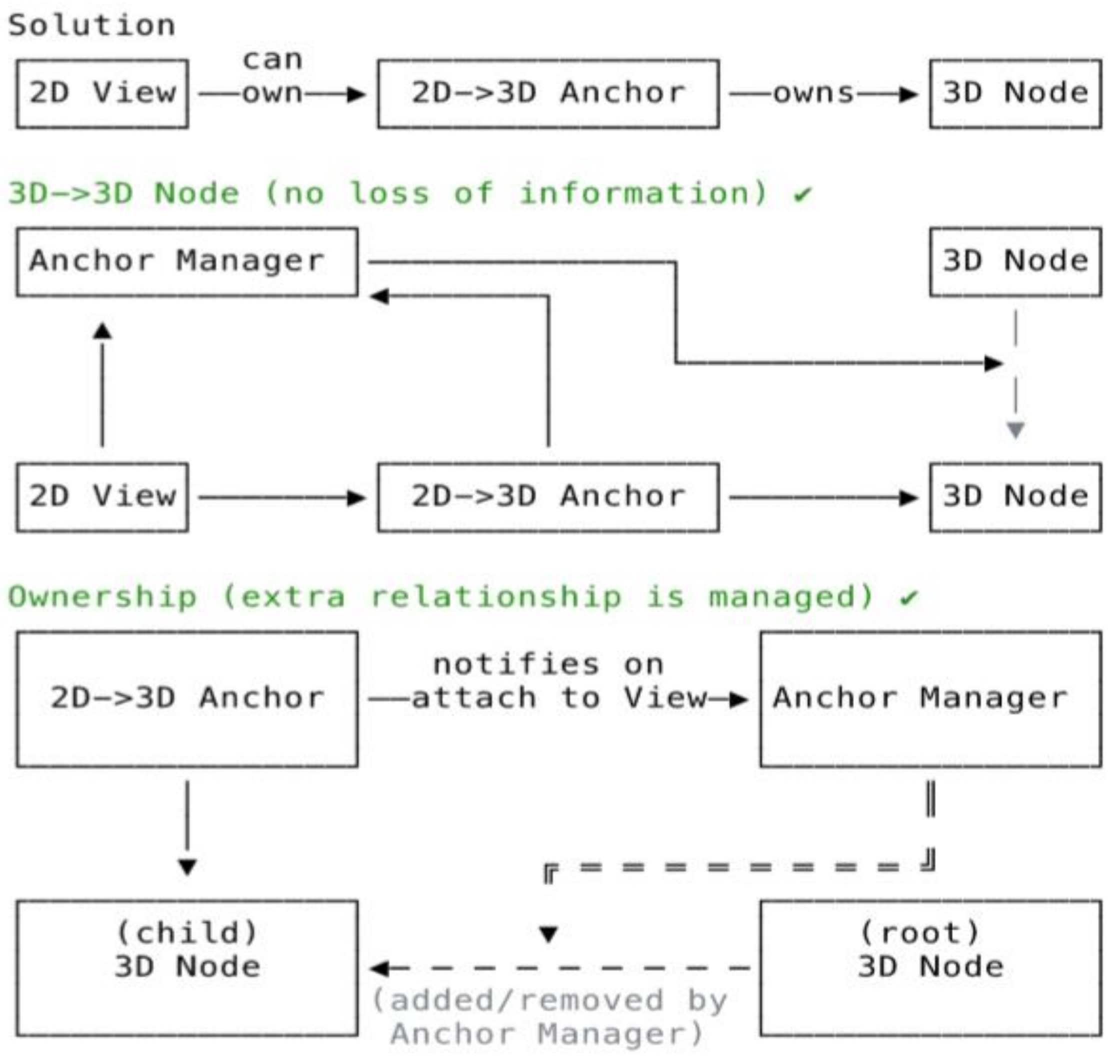

Figure 3: Recovering the information, which is lost by means of anchoring

Additional embodiments

In an additional embodiment, one of the key features of anchoring is a lifetime of the anchor node. The anchor node is attached to the 3D node of the panel as long as the anchor is attached to any of the views. Therefore, the lifetime of the anchor node is exactly same as a lifetime of the anchor. This is ensured by means of the anchor manager. Otherwise, it could lead to subtle bugs if there is a manual management of a relationship between the view and the 3D node or attaching or detaching the 3D nodes incorrectly from their parent 3D node.

In a yet another embodiment, for transforming from the 2.5D environment to the 3D environment, a depth value corresponding to each 2D point may be utilized in a better construction of the 3D environment. A depth map of the 2.5D environment may be generated, which may later be used to account for corrections or remove artifacts in the 3D environment constructed by means of anchoring.

Conclusion

For presenting 3D content on the 2D user interface or to leverage applications of the conventional 2D layout algorithms, the 3D content needs to be transformed into the 2D environment, but this transformation is accompanied with the loss of the information. Similarly, 3D scene reconstruction has tremendous applications ranging from medical diagnosis to computer graphics, virtual reality and so on. Numerous models and algorithms are being developed for generating 3D content or transforming from a lower dimension (i.e. 2D or 2.5D) to a 3D environment appropriately as per the requirements. But similar challenges such as deterioration, loss of information, *etc.*, have been observed while reconstructing 3D scenes. With the solution provided in the present disclosure, these challenges can be overcome to a great extent. Using the anchors, regulated by the anchor manager, in the 3D panel, ensures a smooth transition from the lower dimension to the 3D environment without the usual loss of necessary information that accompanies with the transition from the 3D environment to the 2D environment.