HIVE Kernel Locale Facilities

Jim Purbrick

One of the main aims of the HIVE kernel is to be scalable – to support very large virtual environments with large numbers of users. The principle technique used to achieve this is Locales, a method of breaking down a virtual world into separate Environments and allowing the client to choose which Locales it is interested in.

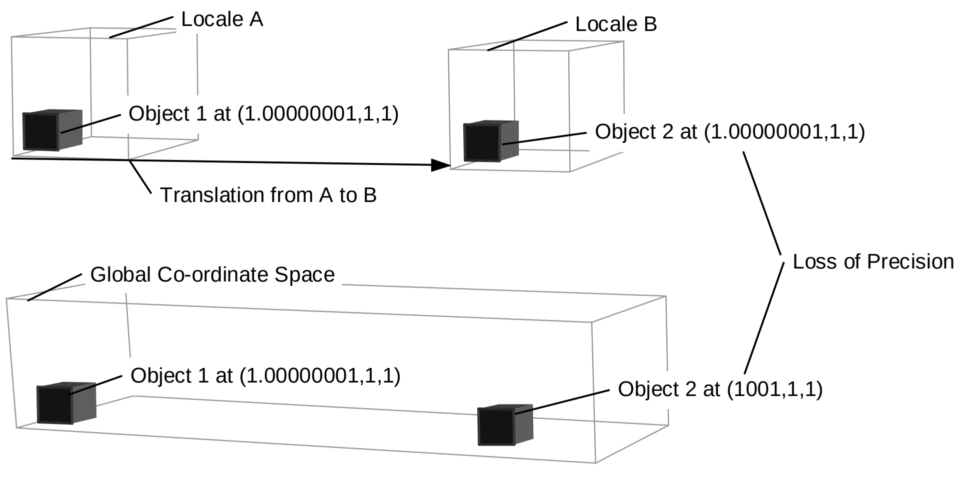
1 Locales

1.1 Geographical Event Filtering

While a virtual world may be very large, most of what a single user can observe and interact with at a given instant is nearby. It is therefore useful for a user to be able to only receive events concerning nearby objects when processing all the events in a world becomes impossible. Locales facilitate this geographical event filtering by dividing the world into separate areas, each of which has its own multicast address. When a user wants to receive events from a Locale it registers itself with the corresponding multicast group. By only joining groups corresponding to nearby locales a user is able to ignore most of the world, most of which is irrelevant.

1.2 Precision

Locales also support large virtual worlds by providing an otherwise impossible level of precision. Each locale has its own local co-ordinate system and the position of an object is only defined with respect to the locale it is in. The relationships between locales are also defined locally so there is no global positioning for the locales. This means that in a multiple locale world, every object can be close to an origin and thus defined with high precision. If a single co-ordinate system is used for the entire world then objects a long way from the global origin are defined at a lower precision.



1.3 Culling, Composition and Interesting Topographies

Aside from providing support for large worlds there are a number of other benefits provided by locales. Because only a small area of the world is joined, the local world model is much smaller than the entire world. This acts as an efficient culling mechanism reducing the number of objects that must be rendered and processed. The framework also lends itself to composing smaller worlds, which can be designed separately and then combined just by defining transformations between the locales. Finally, the mechanism for composing worlds through locales and transforms allows some interesting effects. Making a locale its own neighbour using a reflection transform can create mirrors, while using a translation transform creates toriodal regions. A locale can also be made bigger than that containing it.

2 Locales in SPLINE

The first system to use locales was SPLINE developed at MERL. In SPLINE only the current locale and all locales adjacent to it are included in the world model. This single step replication policy provides simple prediction of a user's movement, as the next locale entered by the user will already be part of the local world model and so need not be added to the model when the locale is entered. This makes locales more transparent to the user than portal based systems like VRML, which builds a local copy of a world as a result of passing through the portal. The portal can not be seen through and the user must wait for the new geometry to be loaded when they pass through.

2.1 Diamond Park

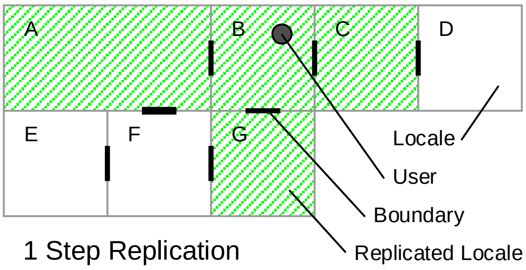
Diamond Park, the demonstration world built for SPLINE, exploits this adjacent neighbourhood strategy in several ways. Small buffer locales are built between larger, more complex spaces so that at any one time only one complex locale is rendered and events from it processed. It includes buildings that are larger on the inside than outside that use vestibule locales to ensure that the inside and outside can not be seen simultaneously. These hide the change in scale and allow the outside and inside to be complex as they will not be rendered together. Similar inconsistent perspectives are used to create transport obelisks that allow a user to quickly move between locales that are usually far apart.

3 Locales in HIVE

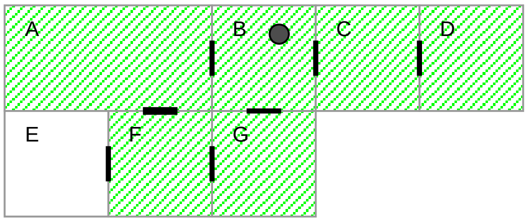
3.1 N Step Replication

The adjacent neighbour strategy used to achieve seamless movement between locales in SPLINE can be seen as a special case of a more general strategy, which replicates all locales within N steps of the current locale. Increasing the size of N allows users to see and interact with more of the world at one time, so allowing different values of N creates a more adaptive strategy. Users on powerful machines or with fast network links can set a higher value to fully utilise their resources, while users struggling with poor frame rates or a congested network can decrease N to concentrate resources on their immediate vacinity.

From an implementation perspective, increasing N above one creates a potential problem. A collection of locales joined by boundaries forms a graph which may be cyclic. In fact generally the graph will be cyclic, as creating a doorway or other reciprocal link between two locales requires setting up two boundaries in opposite directions creating a cycle in the graph. To avoid the cyclic nature of the graph resulting in a locale being rendered multiple times in the same position, Djikstra's algorithm is used to calculate the minimum number of steps needed to reach a locale and build a list of unique locales which can be efficiently rendered. Implementing this method of mapping the locale neighbourhood allowed the addition of a number of other replication and rendering strategies to HIVE.



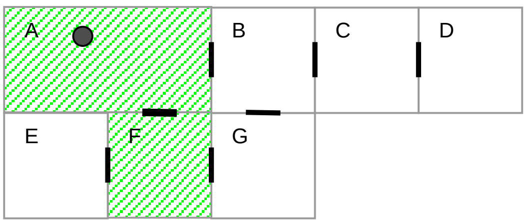
1 Step Replication



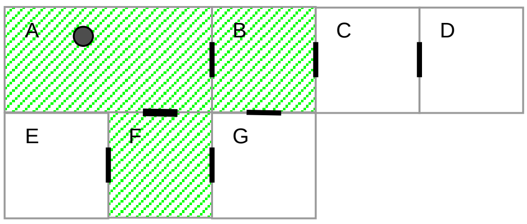
2 Step Replication

3.2 N Nearest Replication

While N step replication can provide a degree of flexibility and performance control, it is very coarse grained – increasing N from 1 to 2 may result in an additional 10 or 20 locales being included in the world model. The obvious improvement to this method is to implement a strategy that includes the N locales closest to the user in the local world model. This both provides a finer grained control over the size of the world model and should maintain it at a more constant size.



2 Nearest Replication



3 Nearest Replication

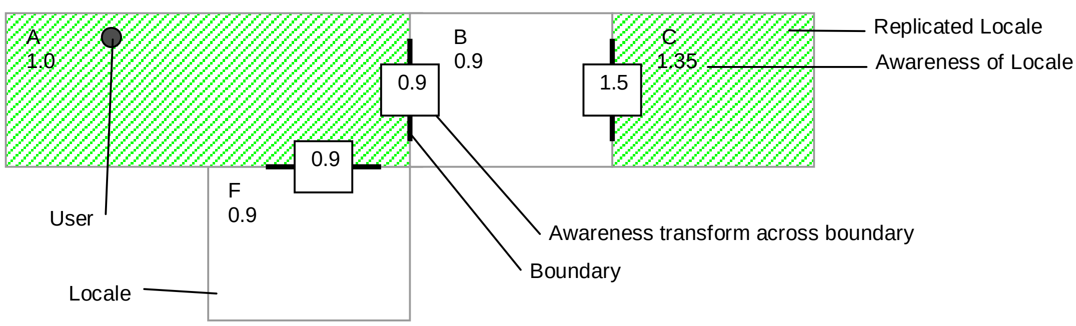
3.3 Awareness Based Replication

Although basing replication policy on prioritising nearby locales is a sensible heuristic, HIVE also includes several replication strategies based on awareness.

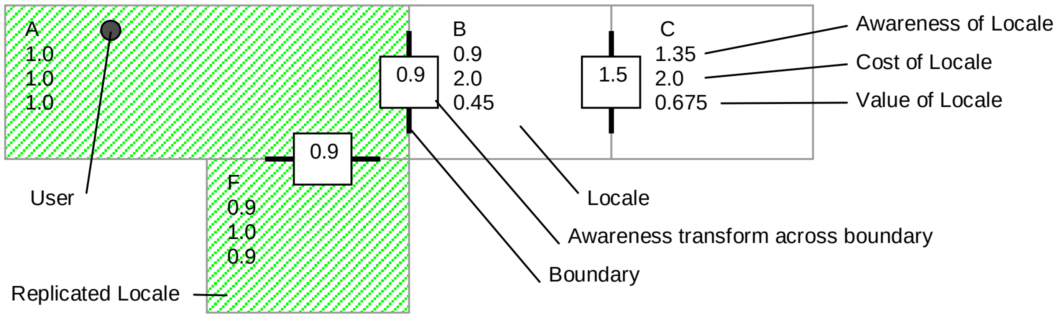
Using this model, each boundary contains a factor by which to multiply the awareness of a user looking through it. A factor of 1 would not change the awareness of objects in the locale on the other side of the boundary and would represent a seamless, open transition across the boundary. A window might have a factor of 0.9 as it reduces visibility slightly, a door a factor of 0 as it is possible to move through, but not see the space on the other side of the door. Video screens or other awareness enhancing objects could be represented as boundaries with an awareness factor of greater than 1. This technique creates a potential catch22 when a locale is being considered for inclusion in the world model: how do we know if the locale contains an object that the user is very aware of without including the locale and examining the objects inside? Currently the awareness policies consider boundaries based on their awareness factor, the idea being that if a boundary severely impairs awareness then even if the objects it contains are directly within the users focus, the user will not be very aware of them. When the spatial model is incorporated into HIVE this can be extended to consider how aware the user is of the boundary as well as the effect of the boundary on awareness. If a user is looking directly at a window it is likely that they want to see through it.

3.4 N Most Aware and Cost/Benefit Replication

The simplest awareness based policy currently implemented in HIVE will replicate the current locale and the N locales with the highest awareness. A more sophisticated strategy is based on a cost/benefit model where each boundary includes a cost value for the target locale as well as an awareness factor. Awareness factor over cost gives the value of the locale and the policy will try to replicate as many high value locales as possible given a budget. The cost is currently a static value that can be used to represent the initial geometric complexity or predicted activity of the target locale. In future it could be dynamically updated to represent the current activity and complexity of a locale which may change as users move around or evolve the locale's geometry.



2 Most Aware Replication

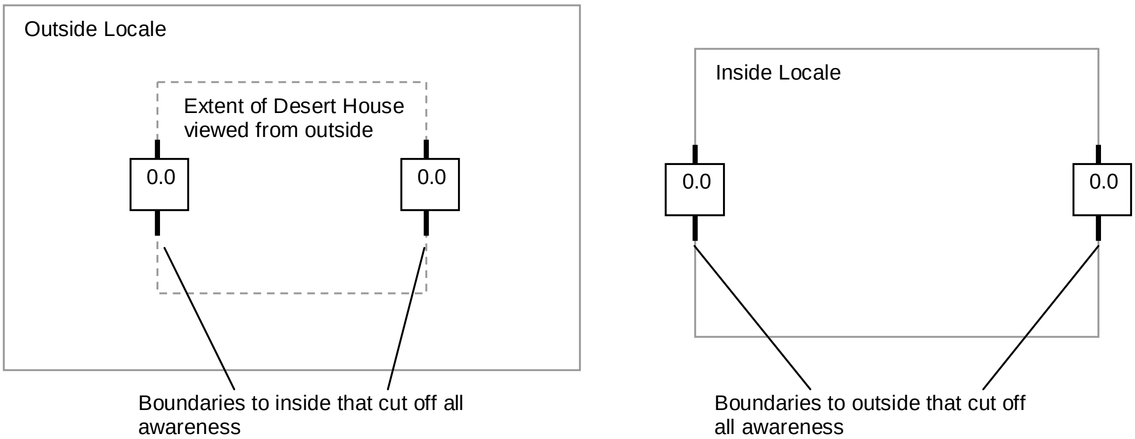


2 Budget Cost/Benefit Replication

Awareness based policies work well for situations were the user is interested in activity a long distance away, like in the HIVE arena environment where the distant playing field is of more interest to a user in the crowd than the other crowd locales nearby. However, if the user then moves to another crowd locale it may not be replicated, it would have to be fetched from a potentially distant location and the user would have to wait. The seamless division of the world would be broken. This suggests that a baseline of nearby locales should be replicated to ensure seamless movement, with any remaining resources spent replicating high value or high awareness locales.

3.5 Diamond Park with HIVE

The more sophisticated replication policies in HIVE are far less reliant on the make up of the world to achieve sensible replication. In Diamond Park many of the benefits of locales are achieved through artifice on the part of the world designers rather than the SPLINE system itself. For example the desert house in Diamond Park is a building with a larger interior than exterior and both the inside and outside locales are geometrically complex. It is important that the user cannot see the two inconsistent scales and desirable that the two complex environments are not rendered at the same time. To achieve this using SPLINE, vestibule locales were added to limit lines of site and provide buffer regions that keep the complex locales 3 steps apart so that SPLINES 1 step replication policy does not keep both locales in the world model together. To achieve the same result in HIVE, normal boundaries can be placed between the outside and inside locales with an awareness factor of 0 and the cost/benefit replication policy used. Whichever locale the user is in, the locale on the other side of the boundary will have 0 value and so will not be replicated by the system or seen by the user. The desired effect is achieved without requiring world designers to know the inner workings of the HIVE system. The awareness factor semantically describes the desired effect rather than achieving it through artifice. This enhances the benefits of locales as a world composition framework and makes the possibility of user construction of locales more feasible as the users will not have to know any implementation issues to achieve satisfactory results.



The Desert House in HIVE

3.6 Abstract Visual Representations

Visual abstractions are a simple addition to the HIVE locale facilities that allow a user to see some indication of a locale even if it is not part of the local world model. The boundary specifies some geometry data and a position for the visual abstraction, which is rendered if the target locale is not replicated. This allows the user to be aware of the locale and move toward it if they are interested in it. When the user is close enough the locale will be replicated and the user will see the real representation of the locale. The rendering of an abstract representation can be a problem when two inconsistent scales are being used as abstract representations positioned using one of the scales may appear inside a locale based on the other scale. To solve this, HIVE makes a special case of boundaries with a 0 awareness factor and does not render any visual representations on the other side of the boundary. Another interesting use of locales in Diamond Park, the transport obelisks, demonstrates the advantages of visual abstractions well. Using SPLINE, the target locales of the obelisk portals are shown using bitmaps to allow users to see where they were going. Using HIVE the abstract representation of the target locale would be rendered automatically.