# A Hierarchical Level of Detail Optimization Algorithm for Frame Rate Control

Ashton E. W. Mason and Edwin H. Blake

**Abstract**

We present a new hierarchical level of detail optimization algorithm that is predictive so is suitable for frame rate control. Our algorithm is general and makes few assumptions about application. We base our approach on the previously demonstrated equivalence of level of detail optimization to the Multiple Choice Knapsack Problem (MCKP). However we show that this equivalence is broken by the unified simple representations for groups of related objects that are typical of hierarchical scene descriptions, and that the equivalence holds instead for a hierarchical generalization of the MCKP. Our level of detail algorithm is an incremental version of a greedy approximation algorithm for this problem, designed to exploit frame-to-frame coherence. We prove that the greedy algorithm's solution is guaranteed at least half-optimal for a useful subproblem in which more expensive selections always provide diminishing returns. Furthermore we argue that its typical solution is much better than half-optimal.

# 1 Introduction

While traditional culling and level of detail techniques serve to reduce the rendering complexity of typical frames, they serve inadvertantly to *increase* the inconsistency of frame rates by making rendering complexity dependent on the variable visible scene complexity rather than on the constant complexity of the entire scene. In order to guarantee constant frame rates the dependency of rendering complexity on scene complexity must be completely eliminated. *Predictive* level of detail techniques offer one way to accomplish this. They regulate frames rates by basing level of detail selection on rough estimates of the rendering cost of available object representations as well as of their perceptual effect. Their aim is to select, for each frame, the set of available representations that provide the best overall perceptual benefit without exceeding the available frame rendering time.

We propose a level of detail optimization algorithm that is essentially a generalization of the well known predictive approach of Funkhouser and Séquin to hierarchical scene descriptions, and corrects problems we outline with that algorithm and previous attempts at its extension. We show that the Funkhouser-Séquin algorithm is not guaranteed at least half-optimal as they claim, due to a flaw in their level of detail selection criterion. In addition we note that their algorithm is not immediately applicable to hierarchical scene descriptions, due to its foundation on the Multiple Choice Knapsack Problem (MCKP). A key contribution of this paper is the demonstration that the provision of unified simple representations for groups of related objects that

is fundamental to many hierarchical level of detail techniques makes the level of detail optimization problem equivalent instead to a new hierarchical generalization of the MCKP. Our algorithm is effectively an incremental version of a greedy algorithm for this Hierarchical MCKP, whose solution we prove is guaranteed half-optimal in the worst case for a useful subproblem in which more expensive selections provide increased perceptual benefit but with diminishing returns.

We begin in Section 2 with a review of related work. In Section 3 we formally define a generalized hierarchical level of detail description. In Section 4 we introduce the Hierarchical MCKP. In Section 5 we present our greedy algorithm for that problem. In Section 6 we prove the worst case half-optimality of the greedy algorithm's solution, for a restricted subproblem. In Section 7 we present our predictive hierarchical level of detail optimization algorithm. Finally in Section 8 we offer some concluding remarks.

# 2 Background

Funkhouser and Séquin [3] were first to note that level of detail - the provision of object representations at multiple resolutions - could be used not only to reduce the complexity of rendering but also to *limit* it predictively by choosing to render only as much detail as may be rendered in the available time. They note the equivalence of this constrained optimization problem to the NP-complete *Multiple Choice Knapsack Problem*, in which a cost-limited subset of a set of candidate items must be selected so as to maximize their total profit, with the restriction that the items are partitioned

into disjoint types (or *candidate subsets*) and exactly one item must be selected of each type [6]:

Given a set $N$ of $n$ *candidate items*, a partition into disjoint *candidate subsets* $N_1, \ldots, N_r$ of the item set $N$ and a *knapsack*, with

$$p_j = \textit{profit} \text{ of item } j \tag{1}$$

$$w_j = \textit{cost} \text{ of item } j \tag{2}$$

$$c = \textit{capacity} \text{ of the knapsack} \tag{3}$$

maximize

$$z = \sum_{j=1}^{n} p_j x_j \tag{4}$$

subject to

$$\sum_{j=1}^{n} w_j x_j \leq c \tag{5}$$

$$\sum_{j \in N_k} x_j = 1 \quad \forall \quad k \in \{1, \ldots, r\} \tag{6}$$

$$x_j \in \{0, 1\} \quad \forall \quad j \in N \tag{7}$$

$$N = \{1, \ldots, n\} = \bigcup_{k=1}^{r} N_k \tag{8}$$

assuming

$$N_h \cap N_k = \emptyset \ \forall \ h \neq k. \tag{9}$$

With regard to rendering, the items are object representations, each with their own perceptual benefit and rendering cost (as predicted by simple heuristics). Exactly one representation must be selected for each scene object. Funkhouser and Séquin describe

a greedy approximation algorithm for MCKP[1] which considers the candidate items in descending order of *value* (profit / cost), selecting each item if it can be afforded. If the item selected belongs to the same candidate subset (or type) as an item that has already been selected, only the item with greater *profit* is retained. The Funkhouser-Séquin level of detail algorithm is an incremental version of this algorithm that is equivalent as long as more expensive selections provide diminishing returns.

Funkhouser and Séquin claim that their algorithm's solution is at least half-optimal (in terms of total profit) in the worst case. Consider however the MCKP instance in which there are 7 items with profits $\overline{p} = (10, 900, 910, 10, 600, 10, 400)$ and costs $\overline{w} = (1, 100, 700, 1, 500, 1, 400)$, partitioned into $r = 3$ candidate subsets. Candidate subset $N_1$ contains items 1, 2 and 3, $N_2$ contains items 4 and 5, and $N_3$ contain items 6 and 7. The capacity of the knapsack is $c = 1000$. The solution reached by the algorithm in this instance is $\overline{x} = (0, 0, 1, 1, 0, 1, 0)$ with total profit $z = 930$, while the optimal solution is $\overline{x} = (0, 1, 0, 0, 1, 0, 1)$, with total profit $z = 1900$. This counterexample may be made arbitrarily bad by manipulating the profits of the items.

Being founded on a greedy algorithm for MCKP, the Funkhouser-Séquin algorithm is fundamentally incapable as it stands of catering for unified representations for groups of related objects. Single unified representations for multiple objects would correspond to single items of multiple type, which are not permitted in MCKP. This is

[1] Actually they refer to the *Continuous Multiple Choice Knapsack Problem*, a relaxation of MCKP in which items may be fractionally selected [6] [4]. However their algorithm never selects fractional portions of items for rendering and is therefore really an algorithm for MCKP.

a significant limitation, since hierarchically unified representations are an elegant and natural means to provide consistent efficient low detail renderings for groups of related objects (See for example [1] [5] and [9]).

Maciel and Shirley [5] present a hierarchical level of detail optimization algorithm that is an extension of the predictive approach of Funkhouser and Séquin to hierarchies with unified *impostor* group object representations. However their algorithm employs a different greedy strategy (based loosely on profit rather than value) and has no guarantees at all of solution quality. Furthermore it is non-incremental and must perform a complete greedy optimization for every frame.

Our level of detail algorithm represents both a correction of the shortcomings of the Funkhouser and Séquin algorithm and a robust generalization of their predictive incremental approach to hierarchical scene descriptions. Our algorithm has much in common with that of *ROAM* [**?**], used for frame rate control in the rendering of height-fields described by Binary Triangle Trees. Like *ROAM*, we make use of two priority queues to prioritize scene elements for level of detail incrementation and decrementation. Our approach differs in that we ...

This paper bares similarities to [7], in which we described an earlier algorithm for the same problem. In this paper we correct an error in that algorithm's detail selection heuristic[2] that caused its solution to be less than half-optimal in the worst case. In Section 6 we provide a formal proof of the correctness of our new algorithm.

[2] A hierarchical version of that afflicting the Funkhouser-Séquin algorithm, which we discovered only after [7] had gone to print.

# 3 Hierarchical Level of Detail Description

Here we define a generalized hierarchical level of detail scene description which will serve as the basis for the following sections. An *object* is defined recursively as the union of other smaller objects which are its *parts*, or children. The hierarchy of objects forms a part-whole decomposition of the scene from a single *scene object* at the root to the smallest, indivisable components at the leaves (see Figure 1). Each leaf object has a number of associated *impostors*, or drawable representations.[3] In addition *group* (or non-leaf) objects may also each be provided with their own set of impostors. An impostor of a group object serves as a unified drawable representation of all the parts of the group. In this way each object has as its drawable representations not only its own explicitly associated impostors at various levels of detail but also the multiple more detailed combinations of the impostors of its descendents. Together these representations constitute the available levels of detail of that object:

**Definition 1** *Level of Detail*

*A* level of detail $s$ *of an object* $O$ *is a set of impostors* $\{i_1, i_2, i_3, \ldots, i_n\}$ *such that exactly one of the impostors on the path from* $O$ *to each of the leaves of the subtree rooted at* $O$ *is an element of* $s$*.*

For example the valid levels of detail of the scene object in Figure 1 are $\{1\}$, $\{2\}$, $\{4,5,3\}$, $\{4,6,3\}$, $\{4,5,7,8\}$ and $\{4,6,7,8\}$. Each constitutes a complete and unambiguous representation of the scene.

[3] We use *impostor* in its most general sense, referring to any drawable object representation [5].

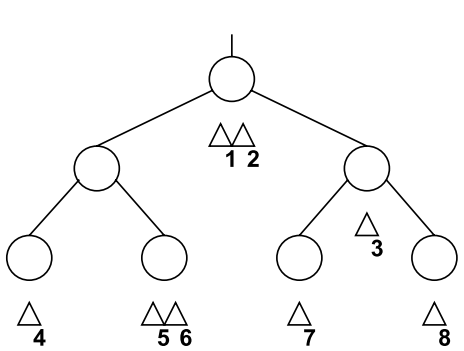


Figure 1: **A simple level of detail hierarchy.** Objects are represented by circles and their impostors by triangles. The impostors of each object are shown in order of increasing detail from left to right. Impostors are labeled arbitrarily for convenience.

**Definition 2** *Replacement Set*

*The* replacement set *of an impostor belonging to an object $O$ is the immediately higher detail impostor of $O$, if one exists, or the set of the lowest detail impostors of the nearest impostor-bearing descendents of $O$, otherwise.*

In Figure 1, the replacement sets of impostors 1, 2, 3 and 5 are $\{2\}$, $\{4, 5, 3\}$, $\{7, 8\}$ and $\{6\}$ respectively. Impostors 4, 6, 7 and 8 have no replacement sets.

An *incrementation* of a level of detail $s$ of an object $O$ is the replacement of some impostor $i \in s$ by its replacement set $r$ to produce another level of detail $s'$. Conversely a *decrementation* of $s$ is the replacement of some complete replacement set $r \subset s$ by the impostor whose replacement set is $r$. The levels of detail of each object are partially ordered by the following relation:

**Definition 3** *Partial Ordering of Levels of Detail*

*Two levels of detail $s$ and $t$ of an object $O$ are related by $s \leq t$ if there exist levels of detail $l_1, l_2, l_3, \ldots, l_n$ such that $l_1 = s$, $l_n = t$, and $l_{i+1}$ is the result of some incrementation of $l_i$ for all $i \in \{1, 2, 3, \ldots, n-1\}$.*

We say that $s$ is a *lower* level of detail than $t$ and that $t$ is a *higher* level of detail than $s$. If $s \leq t$ and $s \neq t$ then we say that $s$ is a *strictly lower* level of detail of $O$ than $t$, denoted $s < t$. For example $\{2\} < \{4, 5, 3\} < \{4, 6, 3\}$ in Figure 1. The *lowest* and *highest* levels of detail of an object are those such that there exist no other levels of detail that are lower and higher, respectively.

**Definition 4** *Ancestor Replacement Sets*

*A replacement set $r$ is an* ancestor *replacement set of another replacement set $q$ if there exists a (possibly trivial) list of replacement sets $r_1, r_2, r_3, \ldots, r_n$ such that $r_1 = r$, $r_n = q$, and $r_{i+1}$ is the replacement set of some impostor in $r_i$ for $i \in \{1, 2, 3, \ldots, n-1\}$.*

Conversely $r$ is a *descendent* replacement set of $q$ if $q$ is an ancestor replacement set of $r$. In Figure 1, $\{3, 4, 5\}$ is a descendent replacement set of $\{2\}$ and an ancestor replacement set of $\{6\}$ and $\{7, 8\}$.

# 4 Hierarchical Multiple Choice Knapsack Problem

In this section we provide a transformation of the hierarchical level of detail description defined in Section 3 to an equivalent *constrained* non-hierarchical one. This transfor-

mation allows us to rigourously interpret the meaning of unified group object representations in terms of the MCKP paradigm.

In the hierarchical level of detail scene description defined in Section 3, each group (or non-leaf) object is the union of its parts, or children. Therefore impostors of group objects are effectively unified representations of all of the parts of those group objects. By our definition they function as lower detail representations of those parts than any of the impostors that are explicitly associated with the parts themselves. We may therefore redraw the hierarchy equivalently by transforming group object impostors into shared low-detail impostors of their children, as long as we note that the shared impostors are constrained and must be selected in unison for all of the parts, if at all.

By repeatedly transforming group object impostors to inherited shared impostors of the children of those group objects, we can create an equivalent *constrained non-hierarchical level of detail description*, as shown in Figure 2. Each object in this new description corresponds to a leaf object in the original hierarchy and has as its impostors all those that lay on the path from the root object to itself, in top-down order. Notice for example that impostor 1 in Figure 2, being an impostor of the scene object, is implicitly a representation of every leaf object. The hierarchical set of constraints preserves the original structure by requiring that each set of inherited impostors is always selected in unison. A valid level of detail of such a constrained description is a set of impostors such that all constraints are satisfied and exactly one impostor is selected for every object.

We can now show that the hierarchical level of detail optimization problem is

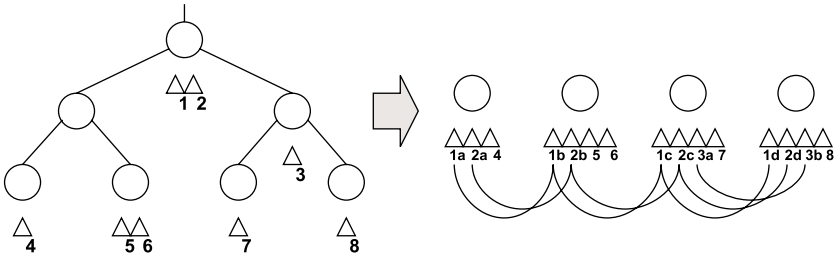


Figure 2: **Transformation of a level of detail hierarchy to an equivalent constrained non-hierarchical description.** Constraints, shown as links, indicate that the inherited shared impostors must be selected in unison. The shared impostors are labelled with letters to distinguish them from one another.

equivalent to a hierarchical generalization of the Multiple Choice Knapsack Problem. In this *Hierarchical MCKP* the candidate subsets are not disjoint; some candidate items are shared between multiple candidate subsets. Each candidate subset corresponds to an object in the constrained non-hierarchical description, and its candidate items correspond to the impostors of the object.

The definition of the Hierarchical MCKP is identical to that of the MCKP given in Section 2 except that line (9) is replaced with a stipulation that the *root item* $o \in N_k, k = 1, \ldots, r$ has a *replacement set* $R_o$. The replacement set of an item $i$ is a set of items $r_i = \{i_1, i_2, i_3, \ldots, i_{z_i}\}$ such that for every candidate subset $N_k$ of which $i$ is an element, there exists exactly one item $j \in r_i$ that is an element of $N_k$; all replacement sets are mutually disjoint; and each item $j \in r_i$ may or may not have a single replacement set.

# 5 Greedy Algorithm for the Hierarchical MCKP

Our greedy algorithm for the Hierarchical MCKP begins with the least expensive feasible solution (corresponding to the lowest level of detail) and iteratively improves on this solution by considering currently selected items for replacement with their replacement sets. In each iteration the item considered is that whose replacement set $r$ has greatest *relative value* $\mathrm{RV}(r)$. The relative value of the replacement set $r$ of an impostor $i$ is defined as the ratio of the difference in perceptual benefit and the difference in rendering cost between $r$ and $i$, ie. $\mathrm{RV}(r) = \frac{(\sum_{j \in r} \mathrm{benefit}_j) - \mathrm{benefit}_i}{(\sum_{j \in r} \mathrm{cost}_j) - \mathrm{cost}_i}$. The incrementation is performed if it can be afforded without exceeding the size of the knapsack, otherwise the greedy selection terminates.[4] After termination, the solution reached is compared against the lowest cost feasible solution containing the *critical replacement set*, which is the first replacement set that could not be afforded. If this solution has greater profit than the greedy selection and has total cost less than or equal to the size of the knapsack then it is selected instead.

# 6 Proof of Half-Optimality

In this section we prove the half-optimality of the greedy algorithm described in Section 5. The algorithm's solution is at least half-optimal (in terms of total profit) for the subproblem of the Hierarchical MCKP in which replacement sets are always more

[4] The algorithm can be trivially improved by continuing to consider any other replacements that might be afforded instead, as long as the items they replace have been selected.

expensive and more profitable than the items they replace, but with diminishing returns for later replacements.[5] Formally, we assume that if $R_j$ is the replacement set of $j$ then

$$p_j \leq \sum_{i \in R_j} p_i, \quad w_j \leq \sum_{i \in R_j} w_i \tag{10}$$

and if $R_k$ is the replacement set of $k \in R_j$ then $\text{RV}(R_k) \leq \text{RV}(R_j)$, ie.

$$\frac{(\sum_{i \in R_k} p_i) - p_k}{(\sum_{i \in R_k} w_i) - w_k} \leq \frac{(\sum_{i \in R_j} p_i) - p_j}{(\sum_{i \in R_j} w_i) - w_j}. \tag{11}$$

Given an instance of the Hierarchical MCKP, let the total profit of the optimal solution to this instance be $z$. Let $G$ be the set of items in the solution reached by the greedy algorithm, and let $z^g = \sum_{i \in G} p_i$ be the profit of this greedy solution.

Then

$$z = z^g + \sum_{j \in A}((\sum_{i \in R_j} p_i) - p_j) - \sum_{j \in B}((\sum_{i \in R_j} p_i) - p_j) \tag{12}$$

where $j$ is the item whose replacement set is $R_j$, $j \in A$ implies that $R_j$ would be selected in the process of selecting the optimal solution but was not selected in the process of selecting $G$ (ie. the algorithm has "underselected" by not selecting $R_j$), and $j \in B$ implies that $R_j$ was selected in the process of selecting $G$ but would not be selected in the selection of the optimal solution (ie. the algorithm has "overselected" by ever selecting $R_j$).

First up we consider the replacement sets $R_j$ such that $j \in A$. When the critical replacement set $R_s$ was considered (and rejected) the set of currently selected items

[5] This ensures that a replacement set with low relative value can never prevent a descendent with higher relative value from being considered.

was exactly $G$. Therefore the critical replacement set $R_s$ was considered as the replacement for some item $s \in G$, and was favoured over some replacement set $R_v$ that is the replacement set of some item $v \in G$ and is an ancestor replacement set of $R_j$. This implies that RV$(R_v) \leq$ RV$(R_s)$:

$$\frac{(\sum_{i \in R_v} p_i) - p_v}{(\sum_{i \in R_v} w_i) - w_v} \leq \frac{(\sum_{i \in R_s} p_i) - p_s}{(\sum_{i \in R_s} w_i) - w_s}.$$

From (11), RV$(R_j) \leq$ RV$(R_v)$. Therefore RV$(R_j) \leq$ RV$(R_s)$, ie.

$$\frac{(\sum_{i \in R_j} p_i) - p_j}{(\sum_{i \in R_j} w_i) - w_j} \leq \frac{(\sum_{i \in R_s} p_i) - p_s}{(\sum_{i \in R_s} w_i) - w_s} \quad \forall \quad j \in A. \tag{13}$$

Next we consider the replacement sets $R_j$ such that $j \in B$. When the critical replacement set $R_s$ was considered for selection (and rejected) the set of currently selected items was exactly $G$. There must therefore exist a list of replacement sets $J_1, J_2, J_3, \ldots, J_z$ such that $J_1 = R_j$, $J_z \subseteq G$, and $J_i$ is the replacement set of some item $j_i$ in $J_{i-1}$ for all of $i = 2, 3, 4, \ldots, z$.

Likewise there must also exist a list of replacement sets $M_1, M_2, M_3, \ldots, M_y$ where $M_1$ is the replacement set of some item in the cheapest feasible solution, $M_y = R_s$ and $M_i$ is the replacement set of some item $m_i \in M_{i-1}$ for all of $i = 2, 3, 4, \ldots, y$.

Then we know that the algorithm at some stage replaced some item $j_z$ in $J_{z-1}$ with $J_z$ instead of replacing some item $m_u$ in $M_{u-1}$ with $M_u$, for some $u \in \{2, 3, 4, \ldots, y\}$, since $J_z$ was selected and $M_y$ (ie. $R_s$) was not. Therefore RV$(J_z) \geq$ RV$(M_u)$:

$$\frac{(\sum_{i \in J_z} p_i) - p_{j_z}}{(\sum_{i \in J_z} w_i) - w_{j_z}} \geq \frac{(\sum_{i \in M_u} p_i) - p_{m_u}}{(\sum_{i \in M_u} w_i) - w_{m_u}}.$$

From (11), RV$(M_u) \geq$ RV$(M_y)$, ie. RV$(M_u) \geq$ RV$(R_s)$. Therefore RV$(J_z) \geq$ RV$(R_s)$:

$$\frac{(\sum_{i \in J_z} p_i) - p_{j_z}}{(\sum_{i \in J_z} w_i) - w_{j_z}} \geq \frac{(\sum_{i \in R_s} p_i) - p_s}{(\sum_{i \in R_s} w_i) - w_s}.$$

From (11) again, RV$(J_1) \geq$ RV$(J_z)$, ie. RV$(R_j) \geq$ RV$(J_z)$. Therefore RV$(R_j) \geq$ RV$(R_s)$:

$$\frac{(\sum_{i \in R_j} p_i) - p_j}{(\sum_{i \in R_j} w_i) - w_j} \geq \frac{(\sum_{i \in R_s} p_i) - p_s}{(\sum_{i \in R_s} w_i) - w_s} \quad \forall \quad j \in B. \tag{14}$$

Therefore, from (12), (13) and (14) we have

$$z \quad \leq \quad z^g + \left[ \sum_{j \in A} ((\sum_{i \in R_j} w_i) - w_j) - \sum_{j \in B} ((\sum_{i \in R_j} w_i) - w_j) \right] \frac{(\sum_{i \in R_s} p_i) - p_s}{(\sum_{i \in R_s} w_i) - w_s}. \tag{15}$$

Let $\overline{c} = c - \sum_{i \in G} w_i$ be the space left in the knapsack after the selection of $G$, immediately before the rejection of the critical replacement set $R_s$. From the fact that $R_s$ was rejected we know the difference in cost between $R_s$ and $s$ is greater than $\overline{c}$:

$$\overline{c} \quad < \quad (\sum_{i \in R_s} w_i) - w_s. \tag{16}$$

Furthermore we know that the total difference in cost between the optimal solution and the greedy solution $G$ must be less than or equal to $\overline{c}$:

$$\sum_{j \in A} ((\sum_{i \in R_j} w_i) - w_j) - \sum_{j \in B} ((\sum_{i \in R_j} w_i) - w_j) \quad \leq \quad \overline{c}.$$

Therefore, from (16),

$$\sum_{j \in A} ((\sum_{i \in R_j} w_i) - w_j) - \sum_{j \in B} ((\sum_{i \in R_j} w_i) - w_j) \quad \leq \quad (\sum_{i \in R_s} w_i) - w_s \tag{17}$$

and so, from (15) and (17),

$$
\begin{aligned}
z &\leq z^g + ((\sum_{i \in R_s} w_i) - w_s) \frac{(\sum_{i \in R_s} p_i) - p_s}{(\sum_{i \in R_s} w_i) - w_s} && (18) \\
&\leq z^g + (\sum_{i \in R_s} p_i) - p_s && (19) \\
&\leq z^g + \sum_{i \in R_s} p_i. && (20)
\end{aligned}
$$

Recall that the greedy algorithm compares the total profit of the final greedy solution (which is greater than or equal to $z^g$) to the total profit $z^s$ of the cheapest solution containing the critical item, and keeps whichever solution is better. That is, the algorithm's solution has profit $z^h \geq \max(z^g, z^s)$. Clearly $z^s \geq \sum_{i \in R_s} p_i$, so $z^h \geq \max(z^g, \sum_{i \in R_s} p_i)$. Therefore, from (20),

$$z^h \geq \frac{1}{2} z$$

and the profit of the algorithm's solution is guaranteed to be at least half the profit of the optimal solution.

# 7 Hierarchical Level of Detail Optimization Algorithm

Our level of detail optimization algorithm is an incremental version of the greedy algorithm described in Section 5. Its advantage is that it exploits frame-to-frame coherence by basing its initial solution on the solution found for the previous frame rather than always performing a complete greedy optimization from scratch. The algorithm is applied once per frame and its input is the rendering cost limit for that frame and the

level of detail selected for the previous frame.[6] Its output is a level of detail for the scene object, the predicted rendering cost of which is guaranteed to be less than or equal to the available rendering time. Upon termination of the algorithm we render the impostors constituting the selected level of detail as the scene representation for that frame.

Like the greedy algorithm, the incremental algorithm is iterative. However in each iteration it both increments the selected level of detail (by replacing a selected impostor by its replacement set) and then repeatedly decrements (by replacing a completely selected replacement set by its associated impostor) while the total rendering cost of the selected level of detail exceeds the rendering cost limit. The impostor selected for incrementation in each iteration is that which can be afforded and whose replacement set has *greatest relative value*. The replacement set replaced in each decrementation step is that with *lowest relative value*. In this way the algorithm greedily adjusts its solution, favouring the selection of replacement sets with high relative value and the deselection of those with low relative value. The algorithm terminates when it finds upon completing an iteration that the replacement set selected in the incrementation step of that iteration was subsequently deselected in one of the decrementation steps.

Conveniently the greedy and incremental algorithms are equivalent as long as (10) and (11) hold. Note that the algorithm as described does not consider the critical replacement set solution (Section 5). The critical replacement set is that which is both

[6] Or in fact any valid level of detail, for example in the case of the first frame.

selected and deselected in the final iteration of the algorithm. However we feel that in practice situations in which the critical replacement set contributes significantly to the optimal solution are unlikely to arise.

Note that the maximum error of the greedy algorithm is bounded by the difference in profit between the critical replacement set and the item it replaces (19). Therefore as the granularity of the candidate items with respect to the knapsack becomes finer, the maximum error of the algorithm tends to zero. In practical level of detail applications the performance of both algorithms can be expected to be much better than half-optimal.[7]

Note also that diminishing returns (as required by (11)) are the norm in level of detail rendering rather than the exception. Successive increases in the complexity of an object representation typically result in progressively smaller improvements in visual quality. Therefore the algorithm can be expected to perform well in practical applications. Care must nevertheless be taken to ensure that the benefit and cost prediction heuristics, being simple ad hoc approximations, do not violate (10) and (11).

The worst case time complexity of both algorithms is $O(n \log n)$ in the number of impostors. However the *average* case complexity of the incremental algorithm depends heavily on frame-to-frame coherence, and our experimental results suggest it is typically closer to $O(n)$ [8]. Pathalogical situations are those in which successive solutions are strongly dissimilar, for example frames with large changes in viewing

[7] A similar observation is made for other Knapsack Problem heuristics in [2].

direction. High frame rates can help to prevent this. Nevertheless the execution time of the algorithm itself is irregular and may occasionally consume a significant portion of the frame time, leaving a shortened portion for actual rendering. These problems can be effectively circumvented by truncating the execution of the algorithm when a set allocated optimization time is up. The nature of the algorithm is such that all of its intermediatery states are feasible solutions. This approach results in brief deterioations of image quality which is quickly regained during times of greater coherence, and can result in extremely regular frame rates.

# 8 Conclusion

We have presented a level of detail optimization algorithm that is a robust hierarchical generalization of the constrained optimization approach of Funkhouser and Séquin to allow unified representations for groups of related objects. This saves rendering complexity while maintaining consistency and conforms to the natural hierarchical description of typical scenes. Our algorithm is predictive and so is suitable for active frame rate control. Its complexity is $O(n \log n)$, and moreover it exploits coherence by basing its initial solution on the solution from the previous frame. The algorithm is essentially a greedy approximation algorithm for a new hierarchical generalization of the Multiple Choice Knapsack Problem. We proved that its solution is always at least half-optimal for a subproblem of the Hierarchical MCKP in which more detailed renderings provide improved image quality with diminishing returns.

# References


[1] B. L. Chamberlain, T. DeRose, D. Lischinski, D. Salesin, and J. Snyder. Fast rendering of complex environments using a spatial hierarchy. In *Graphics Interface '96*, 1996.

[2] M. L. Fisher. Worst-case analysis of heuristic algorithms. *Management Science*, 26(1):1–17, January 1980.

[3] T. A. Funkhouser and C. H. Séquin. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. In *Computer Graphics Proceedings Annual Conference Series*, volume 27, pages 247–254. ACM SIGGRAPH, August 1993.

[4] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[5] P. W. C. Maciel and P. Shirley. Visual navigation of large environments using textured clusters. In *1995 Symposium on Interactive 3D Graphics*, pages 95–102, April 1995.

[6] S. Martello and P. Toth. *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley and Sons Ltd., 1990.

[7] A. E. W. Mason and E. H. Blake. Automatic hierarchical level of detail optimization in computer animation. In D. Fellner and L. Szirmay-Kalos, editors,

*Computer Graphics Forum, proceedings of Eurographics '97*, volume 16, pages 191–199. Eurographics, Blackwell Publishers, 1997.

[8] S. Nirenstein, S. Winberg, A. Mason, and E. Blake. Hierarchical level of detail optimization for rendering of radiosity scenes. Technical Report CS99-02-00, Department of Computer Science, University of Cape Town, 1999.

[9] J. Shade, D. Lischinski, D. H. Salesin, T. DeRose, and J. Snyder. Hierarchical image caching for accelerated walkthroughs of complex environments. In *SIGGRAPH'96, Computer Graphics Proceedings, Annual Conference Series*, pages 75–82. ACM SIGGRAPH, August 1996.