Chapter 4

Analysis

In this chapter we will analyse the vector radiosity approach presented in **Section 3.4** in more detail. We will look more closely at the role the sum-area normal plays in face cluster radiosity, and investigate its relationship to more conventional measures of the projected area of a cluster. We will also develop methods for bounding the transfer of radiosity between two face clusters.

4.1. Sources of Error in Face Cluster Radiosity

I will start by examining possible sources of error in vector radiosity; any approximations made in the method are only acceptable if we have a reliable way of detecting the error in these approximations. As we shall see, it turns out the projected area of a cluster plays a crucial part in such approximations. I will introduce some useful definitions for later sections, and end with an example to illustrate some of the concepts involved.

4.1.1. Recap

In vector radiosity, we are trying to approximate the transfer of radiosity between the individual elements in two clusters of faces, as depicted in **Figure 37**, with a single cluster-to-cluster transfer. We assume the faces within a cluster are all part of one, manifold surface.

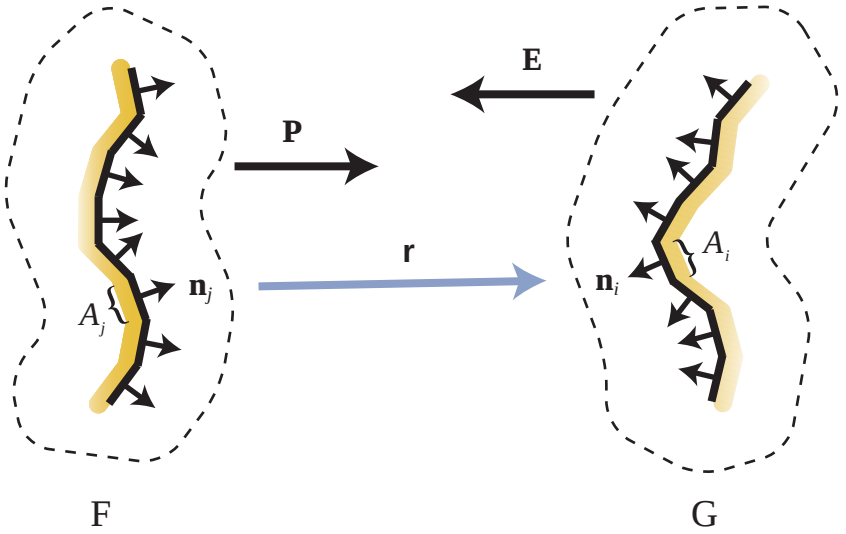


Figure 37: Radiosity transfer between clusters of faces.

As discussed in **Section 3.4.2**, we wish to calculate quickly

$$E_i = \sum_j \frac{(-\hat{\mathbf{n}}_i^{\mathrm{T}}\hat{\mathbf{r}}_{ij})_+(\hat{\mathbf{r}}_{ji}^{\mathrm{T}}\hat{\mathbf{n}}_j)_+}{\pi r_{ij}^2} v_{ij} A_j b_j . \tag{38}$$

We do this by making the vector radiosity approximation:

$$\begin{aligned} E_i &\approx \left(\hat{\mathbf{n}}_i^{\mathrm{T}}\left[\frac{-\hat{\mathbf{r}}}{\pi r^2}(\hat{\mathbf{r}}^{\mathrm{T}}\mathbf{S}_F)_+ b_F \bar{v}\right]\right)_+ \\ &= (\hat{\mathbf{n}}_i^{\mathrm{T}}\mathbf{E})_+ \end{aligned} \tag{39}$$

For each face cluster we have a bounding box, oriented such that the z axis is the normal of the best-fit plane to the faces within the cluster, and we also store the sum area-weighted normal, $\mathbf{S}$,

$$\mathbf{S} = \sum_i \mathbf{S}_i , \tag{40}$$

where $\mathbf{S}_i = A_i\hat{\mathbf{n}}_i$ is the area-weighted normal of face i in the cluster. The area-weighted average face normal of the cluster is then $\hat{\mathbf{S}}$, and its projected area in

that direction is $\|\mathbf{S}\|$. Note that, although $\mathbf{S}$ is roughly perpendicular to the best-fit plane, it is rarely exactly so.

Finally, we also have the total surface area A of the cluster:

$$A = \sum_i A_i . \tag{41}$$

Given this setup, we can state the sources of error in our approximation to the radiosity transfer between face clusters as follows:

- The assumption that the direction vector $\mathbf{r}$ is constant.
- The assumption that $\sum_i (\hat{\mathbf{r}}^{\mathrm{T}} \mathbf{S}_i)_+ = (\hat{\mathbf{r}}^{\mathrm{T}} \mathbf{S})_+$.
- That any self-shadowing within the face clusters is ignored.
- That the occlusion between the two clusters is assumed to be constant.

4.1.2. The Face Cluster Approximation

We are essentially representing the surface within a face cluster by its area-sum normal $\mathbf{S}$, a directed surface element. That is, we are representing a diffusely reflecting curved surface with a single planar Lambertian reflector. If our assumptions about the distance of the cluster from the light sources illuminating it hold up, then when viewed from the direction of $\mathbf{S}$, our approximation is close to perfect. However, from side on, the approximation is not so good. Whereas the apparent size of a single planar element goes to zero as our viewing angle tends to 90 degrees from $\mathbf{S}$, the apparent size of our curved surface does not. In the worst case of a spherical surface, the apparent size will not change no matter where we view it from.

The worst case for this error is generally at the top level cluster, which contains the entire (connected) surface mesh. The planarity metric of the face clustering algorithm takes account of the orientation of faces, and heavily penalises any surface that is curved, so if we descend a few levels in the face cluster hierarchy, the surface will be split into face clusters that are more planar. As long as we detect this source of error, and force subdivision in areas where it is excessive, we can be sure our algorithm will produce a reasonable approximate answer to the global illumination problem.

4.1.3. The Importance of Projected Area

If we multiply the element-to-element radiosity transfer equation, **Equation 38**, by the area of the receiver, we get the received power on element i from element j, which can be written as

$$\frac{(-\hat{\mathbf{r}}^{\mathrm{T}}\mathbf{S}_i)_+(\hat{\mathbf{r}}^{\mathrm{T}}\mathbf{S}_j)_+}{\pi r^2} b_j, \tag{42}$$

where again $\mathbf{S}_i = A_i\hat{\mathbf{n}}_i$. This received power depends on three quantities: the projected areas of the source and destination clusters along $\mathbf{r}$, and a scalar term that accounts for falloff with distance. When we try to use this equation to approximate more complex situations (cluster-to-cluster interactions, for instance), we must be aware of how accurate our approximations to each of these quantities are. Generally this is easy with the distance term, but more difficult to do with projected area terms.

The differences in apparent size between the actual face cluster and our representation of it discussed above can be more accurately viewed as representing an error in our approximation to the actual projected area of the cluster. We are representing the projected area of a face cluster as $(\mathbf{S}\cdot\hat{\mathbf{m}})_+$, and we must be able to decide when this approximation is a bad one in order to force subdivision and improve the approximation. Ideally, this requires bounding the projected area of the cluster in a particular direction $\hat{\mathbf{m}}$. Stamminger et al. have shown how this can be done for simple objects that are capable of providing a cone of normals, but for face clusters we require a more general solution [Stam97c].

4.1.4. Notation

We will be using the mathematical notation shown in **Table 2** for this chapter. There is also a summary of useful physical quantities in Appendix A.

4.1.5. Projected Area Definitions

We define the *visible projected area* (VPA) of a cluster P as

$$\mathrm{VPA}(P, \mathbf{m}) \equiv \int_V \hat{\mathbf{m}}\cdot d\mathbf{S}, \tag{43}$$

where $V \subset P$ is the subset of the surface points in P that are visible from the direction $\hat{\mathbf{m}}$. This is the quantity that we must try to approximate for our transfer calculations.

Symbol	Meaning
$(x)_+$	Maximum of x and 0
$\lceil x \rceil$	Upper bound on x
$\lfloor x \rfloor$	Lower bound on x
$[x]$	Average value of x: $(\lceil x \rceil + \lfloor x \rfloor)/2$
$\langle x \rangle$	The interval of x: $[\lfloor x \rfloor, \lceil x \rceil]$.

Table 2: Mathematical definitions

We also define the *non-negative unoccluded projected area* (NUPA) of the cluster in the direction $\hat{\mathbf{m}}$ as

$$\text{NUPA}(P, \mathbf{m}) \equiv \int_P (\hat{\mathbf{m}} \cdot d\mathbf{S})_+ . \tag{44}$$

This is simply the projected area over all of P, rather than just the visible regions, V. Because occlusion only ever reduces the amount of surface visible in the cluster, $\text{VPA} \leq \text{NUPA}$.

Finally, we define the *signed unoccluded projected area* (SUPA) of a cluster as

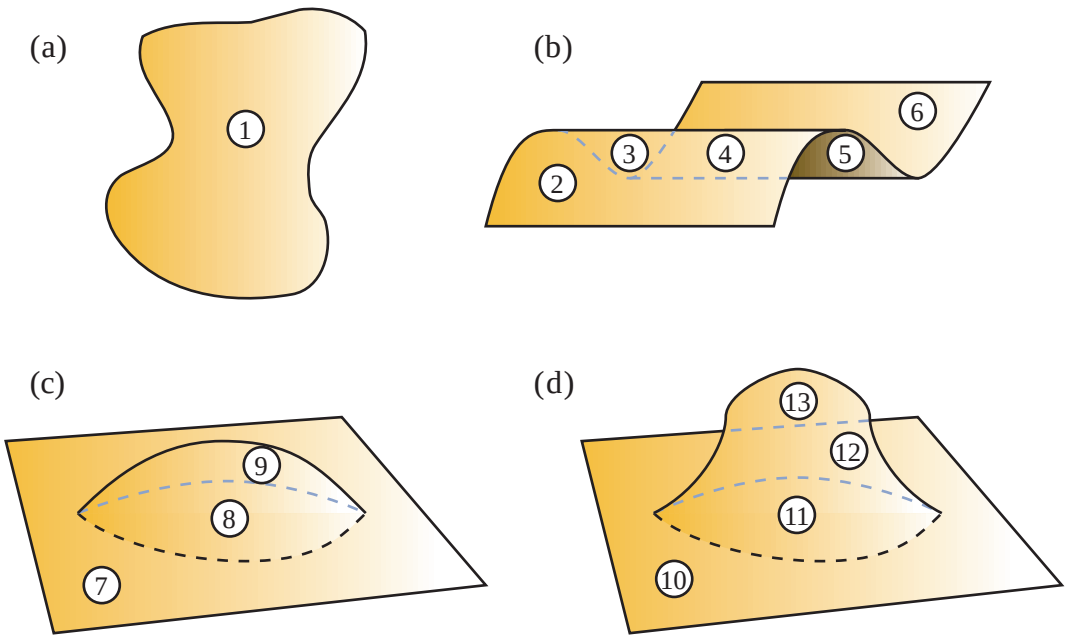
$$\text{SUPA}(P, \mathbf{m}) \equiv ((\hat{\mathbf{m}} \cdot \int_P d\mathbf{S})_+ = (\hat{\mathbf{m}} \cdot \mathbf{S}_P)_+) . \tag{45}$$

For this estimate, we find the sum-area normal vector of the cluster, $\mathbf{S}_P$, and use its clipped dot product with the direction vector $\hat{\mathbf{m}}$ to calculate projected area. This is the equivalent of treating the cluster as a flat surface, and is the primary approximation we employed in **Section 3.4**.

As an illustration of how these methods work, **Figure 38** shows a number of example surfaces split into regions. These projected area of each region may be counted differently by each projected area method. For instance, the VPA will count each region where the surface is facing forwards once, and thus only region 5 is ignored. The NUPA can overcount the forward-facing projected area of some regions, for example region 4, because it ignores visibility. The SUPA can undercount some regions, such as region 11, because the back-facing area in those regions cancels out the forward-facing area.

4.1.6. Visibility Definitions

In our discussions of the role of occlusion in radiosity transfer, it is helpful to establish the following terms:



Surface	Region	VPA	NUPA	SUPA
(a)	1	1	1	1
(b)	2, 6	1	1	1
	3	1	1	0
	4	1	2	1
	5	0	1	0
(c)	7, 8	1	1	1
	9	1	2	1
(d)	10, 11	1	1	1
	12	1	2	1
	13	1	1	0

Figure 38: When calculating the projected area of the above surfaces in the viewing direction, the VPA, NUPA and SUPA count the regions differently, as shown in the table.

- *Tangential visibility*: whether a surface is occluded by its tangential plane. This is usually addressed by clipping the projected area calculation of surface elements to zero.
- *Intra-cluster visibility*: local visibility or shadowing; a surface element is shadowed by other parts of the surface within the face cluster.
- *Inter-cluster visibility*: global visibility or shadowing; the surface element is shadowed by some other surface outside the face cluster.
- *Macro-scale self-shadowing*: the shadowing is on a scale comparable to the surface. For instance, a convex surface where one side shadows the other, or a concave surface where the "lip" of the surface shadows part of the "bowl".
- *Micro-scale self-shadowing*: self-shadowing that occurs on a scale much smaller than the surface. Consider small bumps and notches on the surface. This is the kind of self-shadowing that must be taken into account when computing new BDRFs.

4.1.7. Discussion

An example will illustrate a number of points about dealing with clusters of surfaces. Consider the two-dimensional cluster shown in **Figure 39**, a simple sawtooth surface. In considering the transfer of radiosity between it and a putative source, we must calculate the visible projected surface area of the cluster as seen from that source. From directly above, i.e., source *a*, this will be $\sqrt{2}nA$, where n is the number of teeth in the surface, and A is the area of one side of a tooth. From side-on, source *c*, it will be zero, as only the right-most face is visible, and it is facing away from *c*. From source *b*, which lies in the same direction as the left slope of each saw "tooth", the visible projected area is simply nA.

Traditional volume clustering algorithms deal with such a situation by summing the projected area of each polygon within the cluster in the direction of $\mathbf{m}$. I.e., they find the NUPA of the cluster, $\sum_i (\mathbf{S}_i \cdot \hat{\mathbf{m}})_+$. This has two drawbacks; it requires traversing every polygon within the cluster, and it ignores occlusion of polygons by each other within the cluster. If this intra-cluster occlusion is ignored, the projected area in direction *c* will be calculated as $nA/\sqrt{2}$, rather than zero. This is a large error, and gets worse as n increases. For a largely flat but crinkled surface, it can lead to badly overestimating the amount of radiosity received or radiated from the cluster horizontally.

Some researchers have proposed accounting for intra-cluster occlusion by calculating an isotropic estimate of it. This is done by a Monte-Carlo sampling of

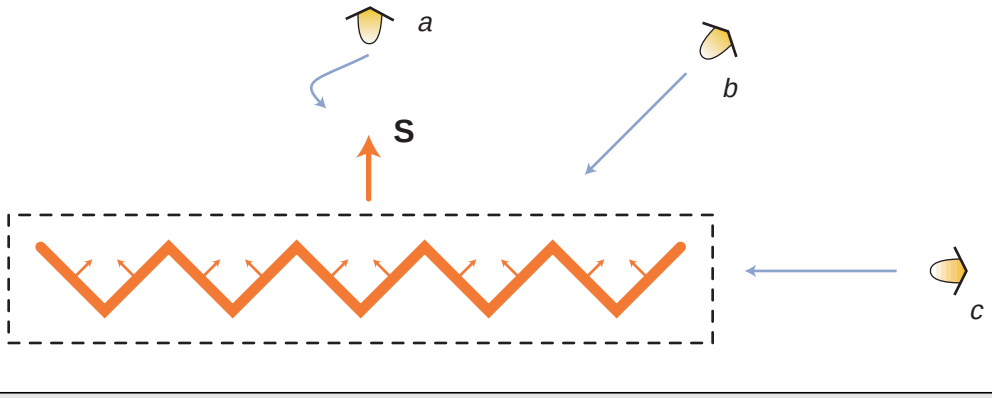


Figure 39: A sawtooth surface. The surface is potentially lit by light sources *a*, *b* and *c*, and has the sum-area normal vector **S**.

the visibility over different points and directions within the cluster to find an average visibility estimate, $\bar{v}$. This may not help much in the case of directional clusters. In the example in **Figure 39**, such an estimate, being isotropic, would result in underestimating the projected area in direction *a*, and would still overestimate the projected area in direction *c*.

It is also possible to calculate an occlusion factor on a per-link basis, in the direction of the link, in the same way that the projected area in the direction of the link is calculated. A number of rays are cast in the direction **m** from points evenly distributed through the cluster; the visibility factor $\bar{v}_{\mathbf{m}}$ is taken to be the average visibility value of these rays. While this improves the estimate markedly, it is expensive, and, in the case of surfaces, it still leads to errors. In the sawtooth case, the average visibility in the vertical direction is $\bar{v}_{\mathbf{m}} = 1/2$, even though all of the surface is visible in that direction, because at least half of the samples will fall behind the surface.

I would argue that, for clusters containing surface meshes, the location of a surface and the visibility of points on that surface are too tightly coupled to be separable in a well-behaved way. There is simply too much coherence in the structure of the surface, and alternative techniques are needed. Ideally we should be sampling visibility from points on the surface contained by the cluster, but tracking such points will either be memory or time-intensive.

The vector radiosity approach we introduced in **Section 3.4** approximates the projected area by storing the sum of area-weighted normals of each polygon

in the cluster as $\mathbf{S}$, and calculating the projected area of the cluster as $(\mathbf{S} \cdot \mathbf{m})_+$. Thus it uses the SUPA estimate of projected area. This reduces the calculation time to a single dot-product, but seemingly also ignores visibility, and is only an estimate.

So, what is the error in approximating the sawtooth surface with a single sum-area normal vector, $\mathbf{S}$? Obviously, from straight above (source *a*), the calculation has no error. From side-on (source *c*), it calculates the projected area as zero, which in this particular case is correct. Also, from an angle from the vertical smaller than the pitch of the surface's "teeth", there is no self-shadowing taking place, and once again $(\mathbf{S} \cdot \mathbf{m})_+$ returns the correct result. Once the angle increases past this, self-shadowing starts to occur, as in **Figure 40**, and to be accurate we must calculate the visible projected area, the VPA, of the surface, taking into account self-occlusion. We might expect that at this point our sum-area normal approximation would fail.

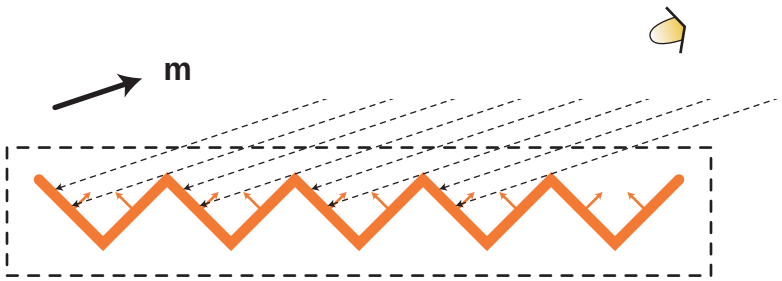


Figure 40: Self-shadowing on the saw-tooth surface. Surprisingly, $(\mathbf{S} \cdot \mathbf{m}_+)$ gives the correct projected area, even taking into account the surface self-shadowing illustrated here.

Surprisingly, this is not the case. It is possible to show with some simple algebra that, regardless of the pitch of the teeth in such a sawtooth surface, its projected area in a direction $\mathbf{m}$, taking into account self-shadowing, is *exactly* $(\mathbf{S} \cdot \mathbf{m})_+$. Thus our estimate of the total projected area of the cluster has turned out to be a much better estimate of the area taking occlusion into account, than the original calculation we were trying to approximate. In this case, the VPA = SUPA.

This result seems almost too good to be true, and unfortunately, in most cases it is. Consider **Figure 41**, which contains an almost identical surface, but one whose side-on projected area is $A/\sqrt{2}$, and not zero as our approximation would predict. It is still the case that the sum-area normal gives a much more accurate estimate of the projected area of the cluster than performing the complete sum, however; the SUPA is a much better approximation to the VPA than the NUPA.

It is also the case that this property does not necessarily hold for weighted area-normal sums. If we instead consider a reflective surface vector, $\mathbf{R} = \sum \rho_i A_i \hat{\mathbf{n}}$, then things work as before if the reflectivity of the surfaces is constant. In **Figure 42** we see a surface where one side of each tooth is considerably brighter than the other. For this surface, **R** underestimates the visible projected reflective area from source *a*, and overestimates it from *b*.

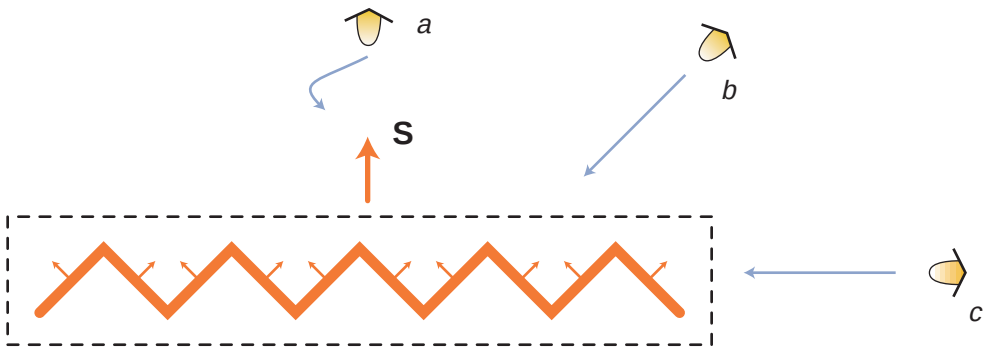


Figure 41: Sawtooth surface with non-zero side area. $\mathbf{S} \cdot \mathbf{m}$ does not give the correct projected area in the direction of *c*.

Our observation raises some interesting questions; what other surfaces might our estimate do a good job on, and under what conditions? It seems that in some situations the sum-area normal can be used to generate an excellent approximation to the visible projected area of a cluster containing a relatively flat surface. But we must formalize this observation for it to be useful.

In **Section 4.3.2** I will outline the conditions under which this observation holds. I will also show how the error of the approximation can be quantified in **Section 4.2**, firstly by proving that the estimate $(\mathbf{S} \cdot \mathbf{m})_+$ is almost always a lower

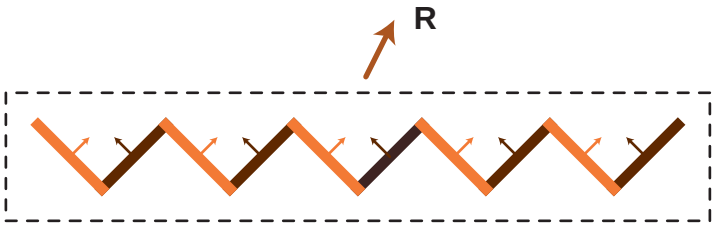


Figure 42: Sawtooth surface with varying reflectance. If we use a reflectance-weighted vector **R**, the estimate fails.

bound on the actual projected area of the cluster, and then by showing how a conservative upper bound can be constructed.

4.1.8. Cancellation

It might seem intuitively that representing a sum of irradiances by a single irradiance vector **E** could lead to cancellation. Consider **Figure 43**, for instance: if we naively sum the irradiance vectors from light sources *b* and *d*, the light from *d* will cancel out the light from *b*, and the point P shown on the surface will be incorrectly illuminated. However this is only a problem if we fail to clip to some consistent plane when calculating **E**. (If we did not have to take into account tangential visibility when calculating the illumination from a light source, then we could indeed just use the vector sum with no error.) To avoid this problem, whenever we gather power across a link, we must clip the resulting irradiance to the corresponding **S**; the same is also true when we push or pull irradiance, as in **Section 3.4.2**.

Even with clipping, it may seem at first glance that light sources can cancel each other out. If we consider light sources *a* and *c* in **Figure 43**, being opposite each other and close to the horizon, it looks as though their horizontal components cancel out when we sum their irradiance vectors. In fact there is no cancellation; this is a standard vector sum. The confusion arises because **E** is implicitly measuring irradiance with respect to the plane of the surface, so the horizontal components of the irradiances from *a* and *c* are irrelevant to the scalar irradiance. Therefore, there is no cancellation problem with our method.

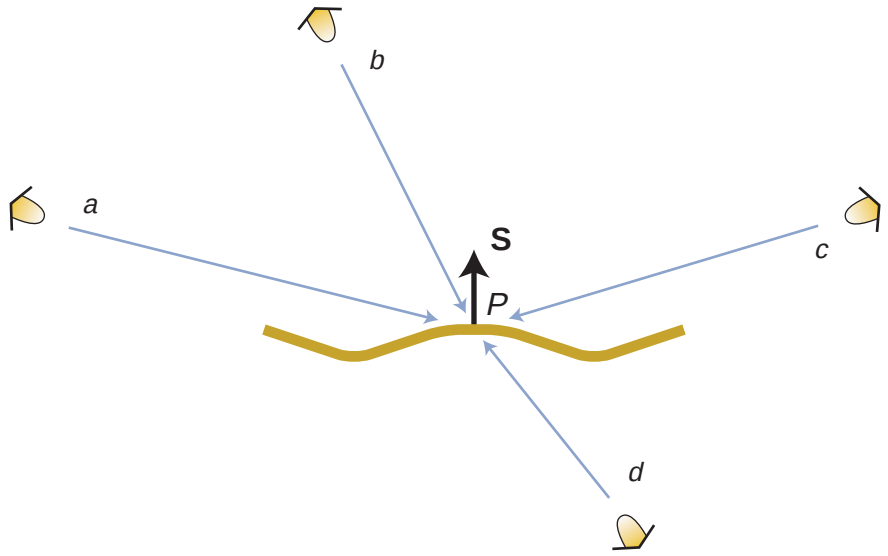


Figure 43: Avoiding cancellation in vector irradiance.

4.2. Bounding the Projected Area of a Cluster

Estimating the actual projected area of a face cluster is a crucial operation in our algorithm. In this section we examine how we can construct bounds on this quantity, such that it can be evaluated in constant time. We start by ignoring visibility, so we are trying to bound the NUPA.

4.2.1. A Lower Bound

Finding a lower bound to the projected area is straightforward. We can use the inequality,

$$(a + b)_+ \leq (a)_+ + (b)_+, \tag{46}$$

to show that, if $\mathbf{S} = \sum_i \mathbf{S}_i$, then

$$(\mathbf{S} \cdot \mathbf{m})_+ \leq \sum_i (\mathbf{S}_i \cdot \mathbf{m})_+ . \tag{47}$$

That is, our approximation to the projected area of a cluster is provably a conservative lower bound on the actual projected area: SUPA ≤ NUPA .

Proof

Obviously given **Equation 46** we have:

$$(\mathbf{S}_0 \cdot \mathbf{m} + \mathbf{S}_1 \cdot \mathbf{m} + \ldots)_+ \leq (\mathbf{S}_0 \cdot \mathbf{m})_+ + (\mathbf{S}_1 \cdot \mathbf{m})_+ + \ldots \tag{48}$$

4.2.2. An Upper Bound

Using **Equation 46** it is possible to show that:

$$\text{if } D_j = \sum_i (\mathbf{S}_i \cdot \mathbf{x}_j)_+, \text{ and } \mathbf{m} = \sum_j c_j \mathbf{x}_j \text{ where } c_j \geq 0, \text{ then}$$

$$\sum_i (\mathbf{S}_i \cdot \mathbf{m})_+ \leq \sum_j c_j D_j. \tag{49}$$

That is, say we sample the projected area of a cluster in a number of different directions $\mathbf{x}_j$. Then, if we can write $\mathbf{m}$ as a linear combination with positive coefficients of some subset of the $\mathbf{x}_j$ vectors, using the same coefficients to interpolate the corresponding projected area samples gives us an upper bound on the projected area in the direction of $\mathbf{m}$. **Figure 44** illustrates how this works in two dimensions.

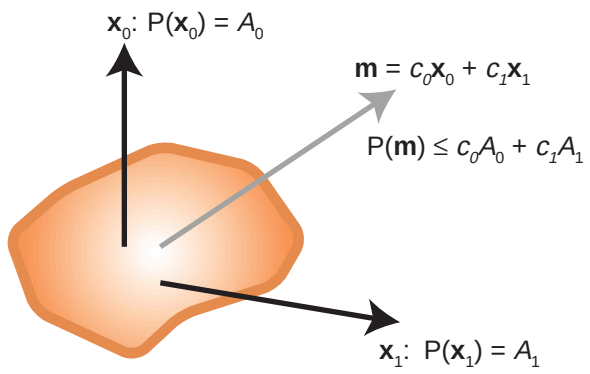


Figure 44: Finding an upper bound from projected area samples.

Neither of these two bounds assume that the surfaces are connected, and thus they can both be applied to volume clusters as well as face clusters.

Proof

We can prove **Equation 49** through the following steps:

$$
\begin{aligned}
\sum (\mathbf{S}_i \cdot \mathbf{m})_+ &= \sum_i \left(\mathbf{S}_i \cdot \sum_j c_j \mathbf{x}_j \right)_+ \\
&= \sum_i \left(\sum_j \mathbf{S}_i \cdot c_j \mathbf{x}_j \right)_+ \\
&\le \sum_i \sum_j (\mathbf{S}_i \cdot c_j \mathbf{x}_j)_+ \\
&= \sum_i \sum_j c_j (\mathbf{S}_i \cdot \mathbf{x}_j)_+ \quad \textbf{if } c_j \ge 0 \\
&= \sum_j c_j \sum_i (\mathbf{S}_i \cdot \mathbf{x}_j)_+ \\
&= \sum_j c_j D_j
\end{aligned}
$$

Implications

If we sample the projected area of a cluster over a number of directions, we can calculate an upper bound on the projected area in an arbitrary direction by interpolating from the three surrounding sampled directions. This estimate will be better the closer to the sampled directions the required direction is, and of course exact if it matches one of the samples[1].

Thus the total projected area of a face cluster can be bounded, as long as we have at least four samples of its projected area in different directions. (The minimal set of samples corresponds to the vertices of the minimal polytope, a tetrahedron.) The fit of the upper bound depends on the sample directions chosen, and the number of samples; obviously the more samples, the better the coverage and resulting bound. Also, if the surface is largely oriented in a particular direction, sampling primarily in that direction will provide better results than sampling in random directions.

One of the simplest collections of directions corresponds to a box, with the projected area sampled in the direction of each of the (six) faces of the box. Inter-

1. I speculate that this technique is approximating the real surface with a polyhedral convex hull.

estingly, if the box in question is a bounding box for the cluster, the six projected-area samples are in most cases bounded by the side areas of the box. There is an important exception; in some cases the fact that we ignore occlusion in the projected area sample can cause it to be larger than the corresponding side area of the box.

A good choice for the basis vectors is the set of axes of the tightest-fit oriented bounding box (OBB) enclosing the cluster. The tightest-fit OBB minimises the side areas of the box, and as the side areas are in turn bounds on the projected area, we are in some sense minimising our upper bounds.

4.3. Intra-cluster Visibility

We now consider the visible projected area of a surface, namely, how to calculate the projected area taking into account intra-cluster visibility. We'll start with some simple illustrative examples.

4.3.1. Self-Shadowing

Radiosity papers commonly ignore self-shadowing by the surfaces under consideration. Even the way the familiar form-factor equation accounts for tangential visibility, by the clamping of both cosine factors to zero, is usually implicit. (Consider the standard radiosity equation in **Equation 22**; it is not explicitly stated that we do not let the cosine factors become negative.) This has partly been because the radiosity method has its roots in the consideration of radiosity transfer between two planar polygons, which inherently have no self-shadowing other than tangential. Because face clusters contain curved surfaces which may self-shadow, we must pay closer attention to this phenomenon.

For mostly planar surfaces, self shadowing is restricted to the micro scale: small folds or bumps in the surface that shadow correspondingly minor parts of the surrounding area. **Figure 45** shows an example. There can also be macro-scale self-shadowing: more major silhouette-type shadowing when the surface is curved, as shown in **Figure 46**, where up to half the surface can be shadowed by the other half, even though the surface itself is relatively smooth.

Interestingly, as discussed in **Section 4.1**, our sum-area normal approximation works well where micro-scale self-shadowing is involved; in some cases it can actually help account for small-scale self-shadowing. An example of this can be seen in **Figure 45**. Using the projected area calculated from $\mathbf{S}$ accounts for the self-shadowing in the small fold shown reasonably well. If we clipped on a per element basis when calculating the projected area of the cluster, we would over-

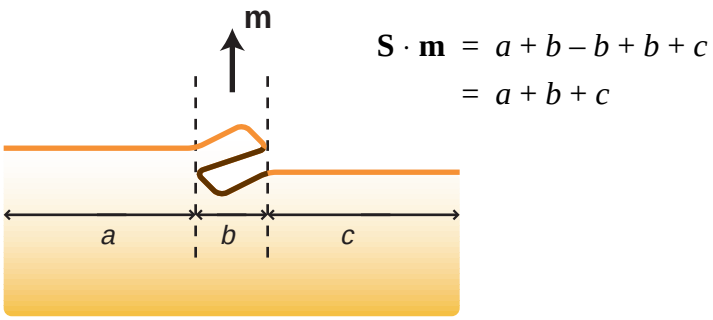


Figure 45: Micro-scale self shadowing.

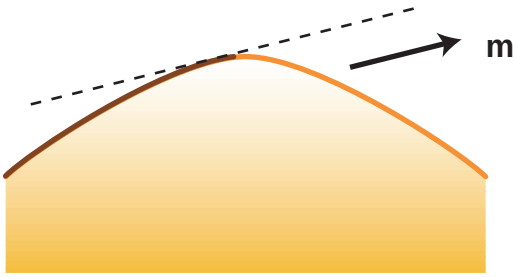


Figure 46: Macro-scale self shadowing.

count contributions from the fold in the surface, and our total projected area in the vertical direction would be $a + 2b + c$.

The way a surface shadows itself is highly correlated with the geometry of the surface itself, and thus its projected area. We need to be able to take it into account, preferably without accessing the detailed geometry of the cluster. (If we don't maintain a constant cost overhead in calculating it, we will lose the efficiency of our algorithm.) Ideally we would like to extend our bounds of a cluster's projected area to bounds on the same area, taking into account self-shadowing. The remainder of this section is thus devoted to the analysis of the visible projected area.

4.3.2. The Visible-Projected-Area Sum Rule

We have observed that in some cases, the sum-area normal approximation to the projected area of *P* is a surprisingly good approximation to the visible projected area. But we need to be clearer about under what conditions this is so. Fortunately, we can construct a rule that allows us to do this. Let us start with the two-dimensional case. The *visible-projected-area rule* for two dimensions is as follows:

> Consider an open path, for which we have defined a top and bottom side; only the top side is considered visible. Consider also the two parallel lines passing through either end point of the path, in direction **m**. We claim that, for the section of the path that falls between these two end-point lines, the visible projected area in direction **m** is exactly equal to the dot product of **m** and the sum of area normals **S**. That is, $\lfloor \mathrm{UPA} \rfloor = \mathrm{VPA}$. Further, we claim that for any remainder of the path lying outside those two lines, the dot product of **m** and the sum of area normals is exactly zero.

Figure 47 illustrates some examples of this rule.

Derivation

There are a number of ways to derive this result. The simplest for purposes of explanation is as follows. We divide the path *P* into slabs pointing in direction **m**, such that each slab has a consistent number of path segments intersecting it. (See, for example, **Figure 47**a-c.) We then connect the two end points of the path with a line *r*, creating a closed path *Q*. We then observe that:

- For any slab, the projected area of a path segment that intersects it is either the positive or negative width of the slab, depending on the orientation of the path segment with respect to **m**.
- The sum projected area in direction **m** of the closed path *Q* within any slab will add to zero. (This is analogous to the famous *winding number* principle used to test for the interior of a closed path.)
- The projected area of the path *P* plus that of the line *r* within each slab is equal to that of the path *Q*, namely zero.
- Within the end-point lines, the projected area of *r* within each slab is the width of the slab; outside it is zero.

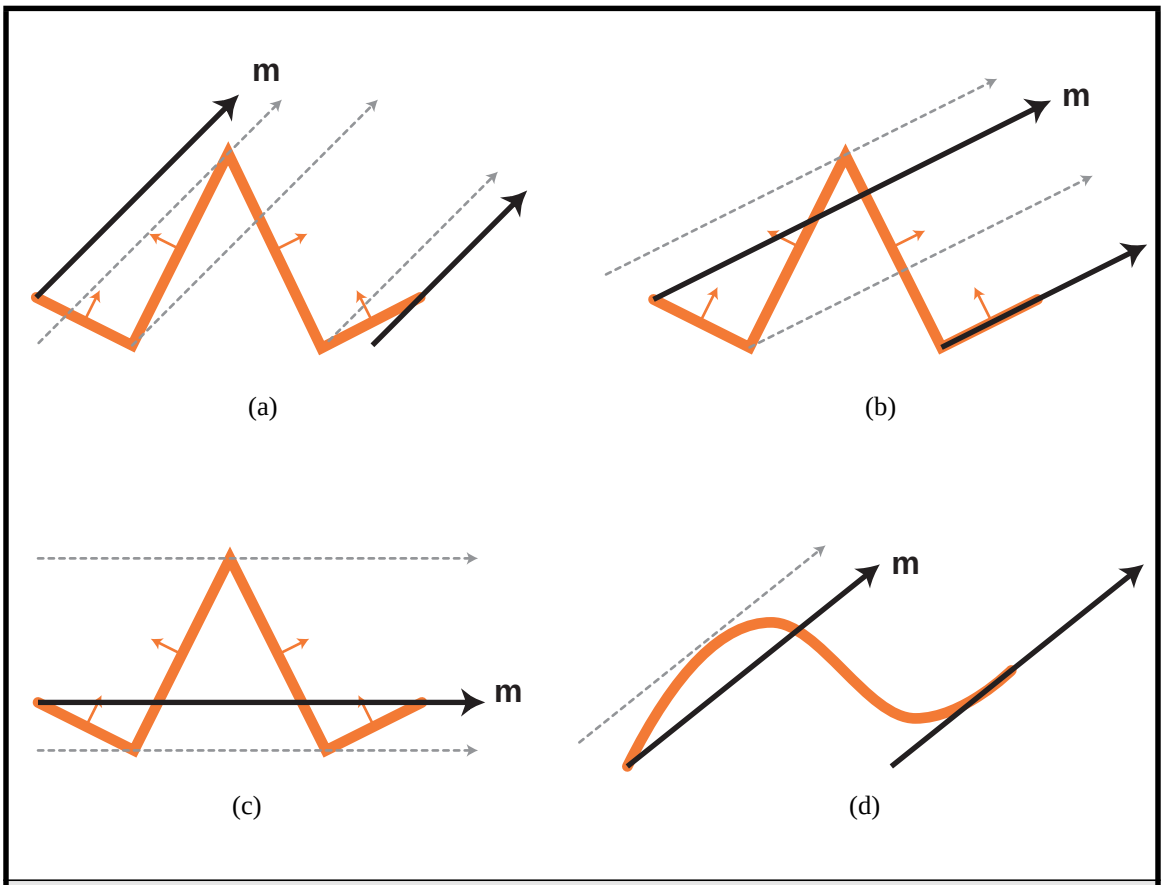


Figure 47: Estimating visible projected areas. In (a) the entire curve falls within the end-point lines, and the estimate $((\sum \mathbf{S}_i) \cdot \mathbf{m})_+$ is correct. In (b), as $\mathbf{m}$ gets more side-on, most of the curve still falls within the lines, but the VPA of the top peak of the curve is incorrectly estimated as zero. (c) Shows the worst case; the end-point lines are coincident, and the visible projected area of the curve is wrongly estimated as zero. (d) Shows how this rule applies to curves, where the estimate is $\int d\mathbf{S}$ rather than a discrete sum.

Thus, for all the slabs that lie between the path end points, the visible projected area of P in direction $\mathbf{m}$ is equal to $(\mathbf{S}_i \cdot \mathbf{m})_+$, where $\mathbf{S}_i \equiv \sum_{\mathbf{S}_j \in slab_i} \mathbf{S}_j$.

For all slabs that lie outside the path end-point lines, this sum is zero. For these slabs, $(\mathbf{S} \cdot \mathbf{m})_+$ obviously underestimates the visible projected area.

A condition on this rule is that the path must not self-intersect. If it does, any path loops cancel themselves out, and visible area can be missed. This is illustrated in **Figure 48**, where using the sum surface vector results in an estimated

projected area of 0; instead the loop in the central slab should contribute a positive amount of area.

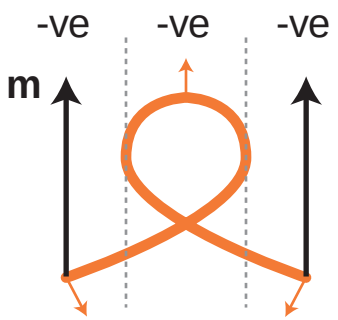


Figure 48: The VPA rule does not hold for paths that self-intersect.

4.3.3. Extension to Three Dimensions

This result can be extended to three dimensions; what was a path becomes a surface, and the end points of the path are now the boundary of that surface. We can state the new rule as follows:

> Consider an oriented manifold surface with a single closed boundary path. Extend the path in the direction **m**, sweeping out an infinitely-long volume V, and thus sectioning the surface into two parts. Then the visible projected area of the surface inside the volume is *in most cases* correctly calculated as $(\mathbf{S}_V \cdot \mathbf{m})_+$, where $\mathbf{S}_V$ is the sum area normal of the surface lying within the volume. $(\mathbf{S}_\Omega \cdot \mathbf{m})_+$, where $\mathbf{S}_\Omega$ is the sum area normal of the surface lying outside the volume, is always zero.

See **Figure 49** for an example. Again, this usually has the same consequence as the two dimensional rule: SUPA ≤ VPA .

There is an important exception to the first part of this rule, hence the qualifiers above. In three dimensions, it is possible for a section of the surface to overlap the rest of the surface without a corresponding back-facing section to cancel out the occluded area. **Figure 50** shows some examples of how this can happen.

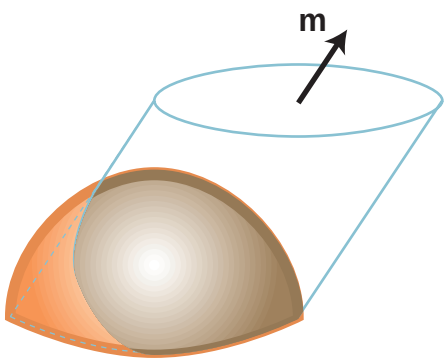


Figure 49: Visible Projected Area Rule in 3D. The surface *P* shown is a hemisphere. When projected in the direction **m**, only the blue region of the surface (to the right), which lies within the open boundary of *P* swept in that direction, is calculated correctly by the SUPA. The external orange region's VPA is incorrectly estimated as zero. As **m** tends to the horizontal, the SUPA underestimates the true VPA more and more.

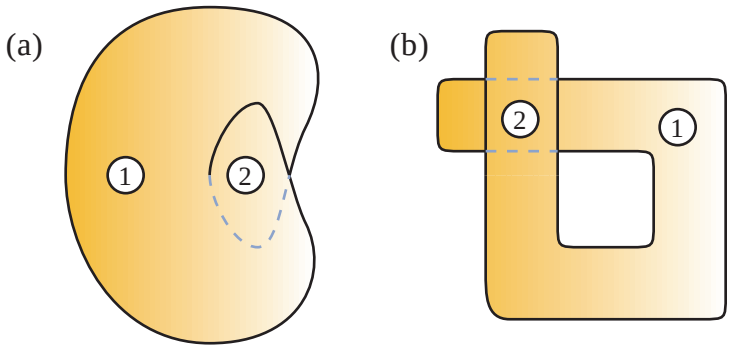


Figure 50: Exceptional cases for the 3D VPA rule. In the examples above the surface overlaps itself in region (2), and the projected surface area in that region will be counted twice, leading to a situation where SUPA > VPA. Fortunately such situations aren't common (see text).

These situations seem to occur rarely in practice. For the models and clustering techniques used in the course of this thesis, the SUPA has proven a good lower bound. (See **Section 4.5.5** for empirical data.) When such overcounting situations do occur, they tend to be for only a small part of the surface, and are often balanced out by undercounting elsewhere.

I surmise, but have not yet proved, that such exceptions cannot happen when the boundary contour is convex. Regardless, it almost always seems to be the case that overlaps correspond to severe concavities in the boundary curve. As our cluster cost metric seeks to produce well-shaped clusters by minimizing boundary length (**Section 3.3.4**, **Section 5.4.3**), it in some sense tries to avoid such situations. In general, the clusters in face cluster hierarchies contain surfaces that are compact and largely concave.

The VPA rule can also be extended to a manifold surface with holes; the accurate calculations will take place for those parts of the surface that are both inside the volume swept out by the boundary contour, and outside the volumes swept out by the hole contours.

4.3.4. Winding Numbers and Boundary Paths

We can be more precise about the conditions under which the SUPA fails as a lower bound on the VPA. In three dimensions it is possible for the projected boundary contour of the surface in direction $\hat{\mathbf{m}}$ to have a winding number greater than one for some regions. It turns out that these regions correspond exactly to possible SUPA / VPA bound violations: Note that each of the overcount regions in **Figure 50** has a winding number of two with respect to the boundary path of the surface.

We can explain this observation as follows. Firstly, we define the winding number of a contour γ with respect to a point p as:

$$w(p, \gamma) = \frac{1}{2\pi} \oint_{\gamma} \frac{1}{l - p} dl \tag{50}$$

Figure 51 shows some examples of different curves and their corresponding winding numbers. We observe that for any connected region that does not cross the contour, all points within will have the same winding number. Thus it makes sense to talk of the winding number of such a region with respect to the contour γ.

Consider the silhouette of a surface projected in direction $\mathbf{m}$. The area within the silhouette is precisely the VPA of the surface. Now, we project the sur-

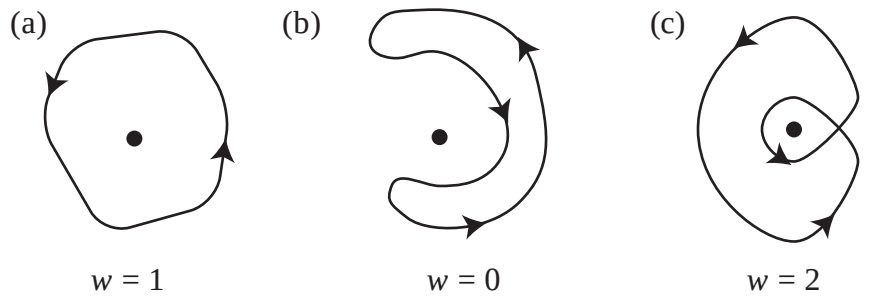


Figure 51: The winding number of a contour. Points inside the projected boundary contour have a winding number of 1, as in (a), and points outside have a winding number of 0, as in (b). When the boundary self-intersects, it is possible to get a winding number greater than 1, as in (c).

face's boundary contour(s) in direction **m**. This projected boundary path must of course lie within the silhouette.

These projected contours will section the silhouette into a set of regions R_i, each of area A_i, such that $\forall p_i \in R_i, w(p_i, \gamma_S) = w_i$. That is, a set of regions such that all points in region i have the same winding number with respect to the projected boundary curve, γ_S. Then, using a trivial application of Stoke's theorem to convert the area integral of the surface sections within each region to the corresponding path integral, we find that

$$\text{SUPA} = \sum_i A_i w_i . \tag{51}$$

The total area of the silhouette is just $\text{VPA} = \sum A_i$.

As a consequence, if the surface is closed, and there is no boundary path, there is just one region of winding number 0, and $\text{SUPA} = 0$. If $\forall i, w_i \leq 1$, then we have $\text{SUPA} \leq \text{VPA}$. If there are regions for which $w_i > 1$, which is only possible if the projected boundary path self-intersects, as in **Figure 51**c, it is possible to have $\text{SUPA} > \text{VPA}$.

An interesting way of looking at the SUPA approximation is:

> For a given direction **m**, we are approximating the *boundary of the projection of the surface* by the *projection of the boundary of the surface*.

That is, we approximate the area within the boundary of the 2D projected surface, its silhouette, by using the area of the projected boundary of the 3D surface[1]. It is possible for areas of the surface to lie outside the boundary path, if it is sufficiently curved; then the SUPA underestimates the VPA. For a flat surface, however, the two are always equal, and $\text{SUPA} = \text{VPA}$.

4.3.5. Implications of the Rule

The visible projected area rule has the following important consequences for face clusters:

- $(\mathbf{S} \cdot \mathbf{m})_+$ is not only a lower bound on the unoccluded projected area of the face cluster, it is *almost always a lower bound on the visible projected area* of the face cluster, except in some uncommon situations where the surface crosses over itself.
- For flat surfaces with mainly micro-level occlusion, using $(\mathbf{S} \cdot \mathbf{m})_+$ to estimate the visible projected area often gives good results.
- The rule means that micro-level self-shadowing is handled well, and macro-level shadowing less well, especially when the surface is highly convex. The worst case comes when the surface is closed, and thus $\mathbf{S} = 0$.

4.3.6. The Bounding Box

Our upper bound from **Section 4.2.1** is a bound on the unoccluded projected area. As occlusion only ever reduces the amount of projected area visible, it does serve as an upper bound on the VPA. However, when there is a lot of concave self-shadowing (as in **Figure 39**), it can fit poorly.

As Stamminger et al. have observed [Stam97c], the projected area of the bounding box of a general cluster is an upper bound on the visible projected area of the entire cluster. To take advantage of its potential superiority in instances of high occlusion, we can combine both bounds. If our directional samples of the projected area correspond with the axes of each cluster's oriented bounding box, we can take

$$D_j^{vis} = min(X_j, D_j), \tag{52}$$

1. Note that the SUPA is in turn just an approximation of this, albeit an exact one when the path does not self-intersect.

where D_j is defined as in **Equation 49**, and X_j is the corresponding side-area of the bounding box. Using D_j^{vis} in **Equation 49** gives us a tighter upper bound on the VPA.

4.3.7. Bounds on the Visible Projected Area

In summary, we have now established bounds on the visible projected area A_P of a cluster P, as:

$$\lfloor \mathrm{VPA}(P, \mathbf{m}) \rfloor \cong (\mathbf{S}_P \cdot \mathbf{m})_+ \tag{53}$$

and

$$\lceil \mathrm{VPA}(P, \mathbf{m}) \rceil = \sum c_j D_j^{vis} \tag{54}$$

where c_j are calculated from $\mathbf{m}$ as in **Section 4.2.2**. We have also shown that usually

$$\lfloor \mathrm{UPA} \rfloor \leq \mathrm{VPA} \leq \mathrm{UPA} \leq \lceil \mathrm{UPA} \rceil, \tag{55}$$

namely, the bounds on the UPA presented in the previous section are often bounds on the VPA.

4.4. Bounding The Radiosity Transfer

A consequence of the hierarchical formulation of the radiosity problem is that for every transfer of radiosity between two parts of the scene that we calculate, we must also calculate an error value that measures the error in the approximation. This error value is used to drive the refinement process.

4.4.1. Choosing an Error Metric

When we evaluate the error in the transfer of radiosity, we must chose some metric with which to do so. Typically we define the norm of a function $f(x)$ as

$$L_n(f(x)) \equiv (\int f(x)^n)^{1/n}. \tag{56}$$

In a radiosity simulation, we are typically measuring the error between our radiosity estimate for a surface, b_i, and the actual radiosity at each point x on the surface, $b(x)$. The choice of an error metric affects both how we calculate our error measure for b_i, and how we combine the radiosity errors of the child nodes in order to calculate the error of their parent. The relevant expressions for both are

shown in **Table 3** for the standard norms (following Lischinski et al. [Lisc94], section 4). In all cases, the error measure is bounded above by an expression of the range of bounds on b_i.

Norm	Error Measure	Parent Error Calculation
L_∞	$\max_{S_i}(\lvert b(x) - b_i \rvert) \le \frac{1}{2}(\lceil b_i \rceil - \lfloor b_i \rfloor)$	The maximum of the children's errors
L_1	$\int_{S_i} \lvert b(x) - b_i \rvert dA \le \frac{A_i(\lceil b_i \rceil - \lfloor b_i \rfloor)}{2}$	Sum the children's errors
L_2	$\sqrt{\int_{S_i} \lvert b(x) - b_i \rvert^2 dA} \le \frac{\sqrt{A_i}(\lceil b_i \rceil - \lfloor b_i \rfloor)}{2}$	Take the root-mean-square of the children's errors

Table 3: Error Norms

Stamminger et al. undertook a study of the results produced by the use of different error metrics in a radiosity algorithm [Stam97b], and concluded that the L_1 and L_2 metrics produced much the same results, and both were better than L_∞. They also found that L_1 and L_2 treat non-conservative bounds better than L_∞. The $\sqrt{A_i}$ factor in the L_2 norm is often taken to be the maximum side length of the polygon being treated; it has been shown that this can be better suited to accounting for the error over long, thin polygons than the straight area-weighted error measure.

Lischinski et al. quote Delves and Mohamed as saying that a cheap, good, occasionally inaccurate error bound often leads to better results than conservative bounds [Lisc94]. This has been borne out in my own experience with bounded radiosity [Will93]. Finding a conservative bound is not enough; it must be shown that as the solution is refined, bounds become more accurate. (We will demonstrate empirically the quality of our bounds in **Section 4.5**.)

The approach to measuring error used in this work is to use the area-weighted irradiance (often referred to as *BFA* in the literature), following the practice of a number of recent hierarchical radiosity papers. Thus the rest of this section will assume the L_1 error norm.

4.4.2. Bounding the Transfer

We can use the bounds we have established on the projected area of a cluster to help bound the transfer of radiosity between two clusters. The bounds give us

both an estimate for the radiosity transfer to use in the simulation, and an error term for the transfer, to use in refining the solution.

The derivation is as follows. The total flux received from a source cluster S by a differential element $d\mathbf{S}_d$ on a destination cluster D is (see **Figure 52**):

$$d\Phi_{d \leftarrow S} = (d\mathbf{S}_d \cdot \int_S b_s v_{sd} \mathbf{m}_{ds} (\mathbf{m}_{sd} \cdot d\mathbf{S}_s)_+)_+ \qquad \text{[Watts]} \tag{57}$$

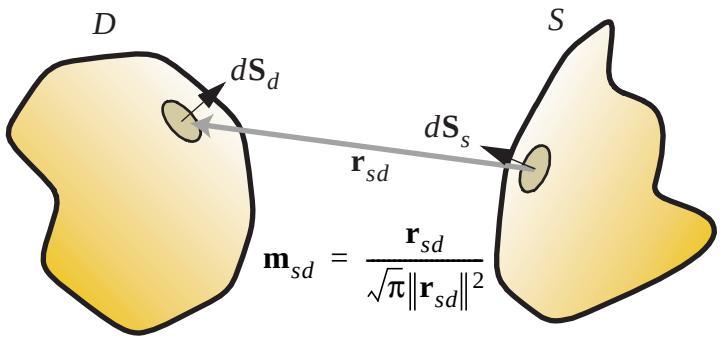


Figure 52: Vector radiosity transfer. Radiosity is transferred between differential surface elements on two face clusters, S and D.

Assume that the source cluster's radiosity is bounded by $\langle b_S \rangle = [\lfloor b_S \rfloor, \lceil b_S \rceil]$ everywhere on its surface. We then wish to formulate an error metric that measures how much error our various approximations are introducing. If we have bounds on the irradiance at every point on the destination cluster, then we can use some L_n norm as a measure of the error in the transfer. As mentioned previously, I use the L_1 norm for this purpose, which corresponds to the *BFA* error metric of traditional hierarchical radiosity, i.e., the error in the total flux Φ (power) received by the destination element. We can then write the flux using interval arithmetic [Moor66, Snyd92, Duff92] as follows:

$$\langle \Phi_{D \leftarrow S} \rangle = \langle G_{D \leftarrow S} \rangle \langle b_S \rangle, \tag{58}$$

where $G_{D \leftarrow S}$ is the geometry term between the two clusters,

$$G_{D \leftarrow S} = \int_D (d\mathbf{S}_d \cdot \int_S v_{sd} \mathbf{m}_{ds} (\mathbf{m}_{sd} \cdot d\mathbf{S}_s)_+)_+ \qquad [\text{m}^2] \tag{59}$$

Thus the total error in the flux can be written as

$$\Delta\Phi_{D\leftarrow S} = \lceil b_S \rceil \lceil G_{D\leftarrow S} \rceil - \lfloor b_S \rfloor \lfloor G_{D\leftarrow S} \rfloor \qquad [\text{Watts}], \tag{60}$$

and the error due to the geometry factor, i.e., the face cluster approximation, as:

$$\begin{aligned} \Delta\Phi^{geom}_{D\leftarrow S} &= [b_S](\lceil G_{D\leftarrow S} \rceil - \lfloor G_{D\leftarrow S} \rfloor) \\ &= [b_S]\Delta G_{D\leftarrow S} \end{aligned} \tag{61}$$

If we were to store bounds on the source cluster's radiosity, we could likewise calculate the error in $\Phi_{D\leftarrow S}$ due to Δb_S, $\Delta\Phi^{b}_{D\leftarrow S}$, at the cost of added memory use.

We calculate the geometric error $\Delta G_{D\leftarrow S}$ when creating a link, and store it with that link. We then take its product with b_S on each iteration, and use the result to decide whether to refine the link.

We can use interval arithmetic to rewrite $\langle G_{D\leftarrow S}\rangle$ as the product of the bounds on the two projected-area calculations, first rewriting it in discretized form:

$$\begin{aligned} \langle G_{D\leftarrow S}\rangle &= \langle \sum_{i\in D} \left(\mathbf{S}_i \cdot \sum_{j\in S} v_{ij}\mathbf{m}_{ij}(\mathbf{m}_{ji}\cdot\mathbf{S}_j)_+ \right)_+ \rangle \\ &= \langle \sum_{i\in D} v_{ij}^{D\leftarrow} (\mathbf{m}_{ij}\cdot\mathbf{S}_i)_+ \rangle \langle v_{ij}^{S\leftrightarrow D} \rangle \langle \sum_{j\in S} v_{ij}^{S\leftarrow} (\mathbf{m}_{ij}\cdot\mathbf{S}_j)_+ \rangle \\ &= \langle A_D^{vis}(\langle \mathbf{m}_{DS}\rangle)\rangle \langle v^{S\leftrightarrow D}\rangle \langle A_S^{vis}(\langle \mathbf{m}_{SD}\rangle)\rangle \end{aligned} \tag{62}$$

where

$$\mathbf{m}_{ij} = \int_{i\in D}\int_{j\in S} \mathbf{m}(d_i \leftarrow s_j) \qquad [\text{m}^{-1}], \tag{63}$$

and $A_D^{vis}(\mathbf{m})$ is the visible projected area of cluster D in direction $\mathbf{m}$, and $A_S^{vis}(\mathbf{m})$ that of cluster S in the opposite direction. The term $v_{ij}^{S\leftrightarrow D}$ accounts for all occlusion that takes place between the two clusters, but is not caused directly by either one, $v_{ij}^{D\leftarrow}$ represents any self-occlusion by cluster D, and $v_{ij}^{S\leftarrow}$ does the same for cluster S.

While we have a way of calculating strict bounds on $A_D^{vis}(\mathbf{m})$ and $A_S^{vis}(\mathbf{m})$ for a single direction $\mathbf{m}$ (from **Section 4.3.7**), we do not have an analytical way to do this for a range of directions, $\langle\mathbf{m}\rangle$. (In this case we are interested in the bounds over the range of directions between the two clusters.) We can instead use the standard Monte-Carlo solution to the problem of estimating these bounds by using an appropriate number of samples of $\mathbf{m}_{sd}$ [Lisc94]; this seems to work well.

4.4.3. Refinement

Given that the bounds on the transfer indicate we should subdivide, which of the source and destination clusters should we pick to subdivide? One way to solve this problem is to decide which cluster contributes most to the error in E. If we have a function $F(x, y)$, and it is separable so that $F(x, y) = G(x)H(y)$, the amount of error due to x and y, respectively, is $\Delta G(x)\overline{H(y)}$ and $\overline{G(x)}\Delta H(y)$. If we ignore the error due to m_{ij}, then, and define

$$Q_i = \sum_j (\mathbf{m}_{ij} \cdot \mathbf{S}_j)_+ \tag{64}$$

and vice versa, then the geometric error due to the source is

$$\Delta G_S = (\lceil Q_j \rceil - \lfloor Q_j \rfloor)\frac{\lceil Q_i \rceil + \lfloor Q_i \rfloor}{2} \tag{65}$$

and vice versa. We then choose the cluster with the largest error to subdivide.

This approach is somewhat more accurate than the traditional “choose the patch with the largest projected area”. It has the advantage that it is more likely to subdivide a cluster that has a small projected area in the direction of the transfer but a large relative error. In particular, it promotes the subdivision of clusters on the silhouette of an object.

4.4.4. Interreflection

We must consider the special case of a radiosity transfer link between a face cluster and itself, the so-called “self-link”. This link measures internal reflection. Traditionally this is calculated by using the same process as a link between two separate clusters. This tends to fail badly for clusters containing any kind of coherent surface. Interreflection is closely tied to self-shadowing, so we can hope to use the bounds on the visible projected area to help.

Most hierarchical radiosity methods ignore micro-scale interreflection. They do this because in its standard form, hierarchical radiosity is a top-down algorithm, in that solution starts at the coarsest representation of the scene, and proceeds to more detailed scene representations as judged necessary; a process known as refinement.

A problem with this approach is that it is hard to estimate the interreflection of a cluster successfully. One approach is to take the amount of form-factor “left over” from 1 after interactions with external nodes are taken into account as an upper bound on the amount of internal interreflection. If the external interac-

tions are small (and this can easily be the case — consider any surface illuminated solely by a small but powerful light source), this can badly overestimate the amount of internal interreflection.

With our visible projected area bounds, we can hope to do better. If we consider a large sphere enclosing a cluster, as in **Figure 53**, then the total form-factor from that cluster to the sphere should account for all possible external interactions the cluster could have. The total internal form-factor is then simply the remainder of this from 1. This is as above, only now we are accounting for all possible external interactions, and not just those that actually exist.

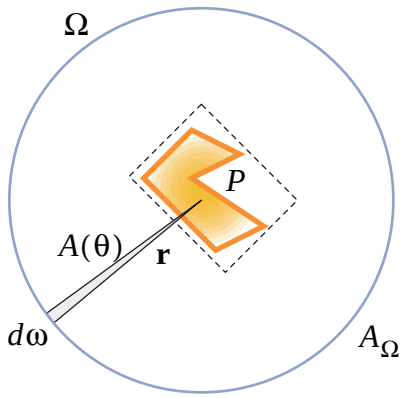


Figure 53: Calculating the external form factor. The projected area of cluster P in direction θ is integrated over the enclosing sphere Ω having surface area A_Ω.

Thus we can write the internal form-factor of cluster P as

$$\begin{aligned} F_{P \leftarrow P} &= 1 - F_{P \leftarrow \Omega} \\ &= 1 - \frac{A_\Omega}{A_P} F_{\Omega \leftarrow P} \end{aligned} \tag{66}$$

and, if the cluster has projected area $A(\theta)$ in direction θ, we have

$$F_{\Omega \leftarrow P} = \frac{A_P^{vis}}{A_\Omega}, \tag{67}$$

where A_P^{vis} is the *visible external area* of P, and is defined as

$$\begin{aligned} A_P^{vis} &= \int_\Omega \frac{A(\theta)}{\pi r^2} r^2 d\omega \\ &= \frac{1}{\pi}\int_{-\pi}^{\pi}\int_0^{2\pi} A(\theta)\cos\theta d\phi d\theta \end{aligned} \tag{68}$$

It is a straight-forward exercise in integration to use **Equation 67** to show that:

- For a planar cluster, $A(\theta) = A_P(\cos\theta)_+$, and thus $A_P^{vis} = A_P$. This gives us $F_{\Omega \leftarrow P} = A_P / A_\Omega$, and hence $F_{P \to P} = 0$.
- For a cluster that radiates evenly in all directions (for instance, a sphere), $A(\theta) = A_X$ (where A_X is the cross-sectional area[1]), $A_P^{vis} = 4A_X = A_P$, and thus again $F_{P \to P} = 0$.

Both of these are as we expect: neither surface features any self-shadowing and thus interreflection.

We can use our lower bound on the visible projected area, $(\mathbf{S}_P \cdot \mathbf{m})_+$, to establish an upper bound on the internal form-factor. We find that, similar to the planar case, $A(\theta) = (\mathbf{S}_P \cdot \hat{\mathbf{n}}_\theta)_+$, $\lfloor A_P^{vis} \rfloor = \|\mathbf{S}\|$, and thus

$$\lceil F_{P \leftarrow P} \rceil = 1 - \frac{\|\mathbf{S}_P\|}{A_P}. \tag{69}$$

If the cluster P contains a completely flat surface, $\|\mathbf{S}_P\| = A_P$ and $\lceil F_{P \leftarrow P} \rceil = 0$ as we might expect.

For the lower bound, it is possible to show that, for three orthogonal vectors $\mathbf{x}_{0\ldots2}$ spanning an octant of $\Re^3$, and with corresponding projected area samples $D_{0\ldots2}$, the integral of the projected area over that octant is:

$$A_{\text{octant}}^{vis} = \frac{\pi}{4}(D_0 + D_1 + D_2). \tag{70}$$

1. The cross-sectional area of a sphere is πr^2; its total surface area is $4\pi r^2$.

Thus, for our standard complete set of six directional samples, the integral over the sphere is

$$\lceil A^{vis} \rceil = \sum_{i=0}^{5} D_i = D_{sum}. \tag{71}$$

If our cluster contains the usual worst-case sphere, **Equation 70** will over-estimate the total surface area as $6D_0$ rather than $4D_0$, i.e., $(3/2)A_\Omega$, leading to a negative lower bound on the internal form-factor; thus we must be careful to clip A_P^{vis} to A_P, so that:

$$\lfloor F_{P \leftarrow P} \rfloor = 1 - \frac{max(A_P^{vis}, A_P)}{A_P}. \tag{72}$$

Using more samples per cluster would help reduce the overestimated visible surface area, but at the cost of added memory. (As our clusters are constructed to be as planar as possible, we think the existing error is acceptable.)

Proof of Equation 70

Assume without loss of generality that our orthogonal sample directions are the *x*, *y* and *z* axes, and that our area samples in those directions are a, b and c respectively. Then

$$\begin{aligned} \lceil A(\hat{\mathbf{v}}) \rceil &= a\hat{\mathbf{v}}_x + b\hat{\mathbf{v}}_y + c\hat{\mathbf{v}}_z \\ \lceil A(\theta, \phi) \rceil &= a\cos\theta\sin\phi + b\cos\theta\cos\phi + c\sin\theta \end{aligned}, \tag{73}$$

and

$$\begin{aligned} A^{vis}_{\text{octant}} &= \int_0^{\frac{\pi}{2}}\int_0^{\frac{\pi}{2}} (a\cos\theta\sin\phi + b\cos\theta\cos\phi + c\sin\theta)\cos\theta\, d\phi\, d\theta \\ &= \int_0^{\frac{\pi}{2}} \left((a+b)\cos^2\theta + \frac{\pi}{2} c\sin\theta\cos\theta \right) d\theta \\ &= \Big\lfloor \frac{1}{2}(a+b)(\sin\theta\cos\theta + \theta) - \frac{\pi}{4} c\cos^2\theta \Big\rfloor_0^{\frac{\pi}{2}} \\ &= \frac{\pi}{4}(a+b+c) \end{aligned} \tag{74}$$

4.4.5. Interreflection of Leaf Clusters

In previous radiosity methods, the total element area was used to calculate a leaf element's radiosity from the incident power, in a calculation similar to **Equation 37**. This works well when the leaf solution elements are always planar. However, problems arise when they are not, and internal interreflection occurs. Previous methods always descended to the input polygons, so that this was not a problem, but in face cluster radiosity our leaf solution elements can be face clusters.

Consider the cluster shown in **Figure 54**, which consists of a small externally-visible region, A_{vis}, and a larger internal region. Externally, it seems the cluster should act as a small flat element of area A_{vis}. On the other hand, we can let the total area of the cluster become arbitrarily large without affecting A_{vis}, by increasing the internal area, and thus decreasing our estimate of the cluster's radiosity. This seems counter intuitive, and leads to the question of how we can reconcile such a situation with the constant radiosity assumption that leads us to divide by total area.

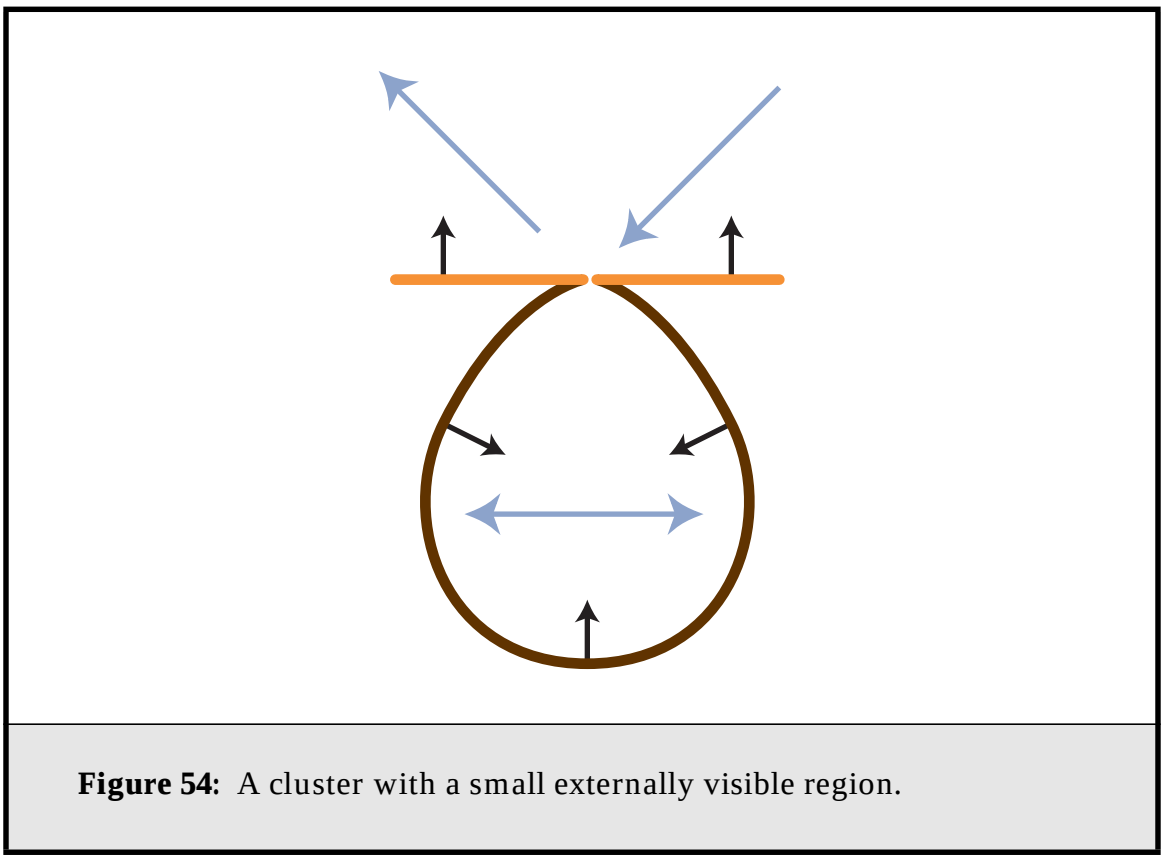


Figure 54: A cluster with a small externally visible region.

The answer is that the difference between our intuition and the calculations we are performing is that we are ignoring potential internal interreflection. To account for this, we must use the internal form-factor of the previous section to write

$$b = F_{P \leftarrow P} b + \frac{\rho \mathbf{S}^{\mathrm{T}} \mathbf{E}}{A}. \tag{75}$$

(We will ignore emittance for the moment.) Solving for b gives us

$$\begin{aligned} b &= \frac{\rho \mathbf{S}^{\mathrm{T}} \mathbf{E} / A}{1 - F_{P \leftarrow P}} \\ &= \frac{\rho \mathbf{S}^{\mathrm{T}} \mathbf{E} / A}{1 - (1 - A_{vis} / A)} \\ &= \frac{\rho \mathbf{S}^{\mathrm{T}} \mathbf{E}}{A_{vis}} \end{aligned} \tag{76}$$

and adding back in emittance gives a new form of **Equation 37**,

$$b = \frac{\rho \mathbf{S}^{\mathrm{T}} \mathbf{E}}{A_{vis}} + e, \tag{77}$$

which satisfies our intuition. The radiosity of the example cluster will be calculated relative to its visible external area, and increasing the internal area of the cluster will have no effect on its radiosity.

Thus, to take into account internal interreflection, we substitute the equivalent cluster external area, A_{vis}, for A. We only do this at the leaf clusters, when calculating the leaf radiosities, because for any other cluster, any interreflection is taken care of by transfer links between its children. Self-links are thus unnecessary for face clusters.

4.5. Empirical Projected Area Results

To test the goodness of fit of our bounds, we ran tests on a number of polygonal models that had been face clustered. Results for two of these objects, a teacup and a pipe fitting (**Figure 55**) are presented here.

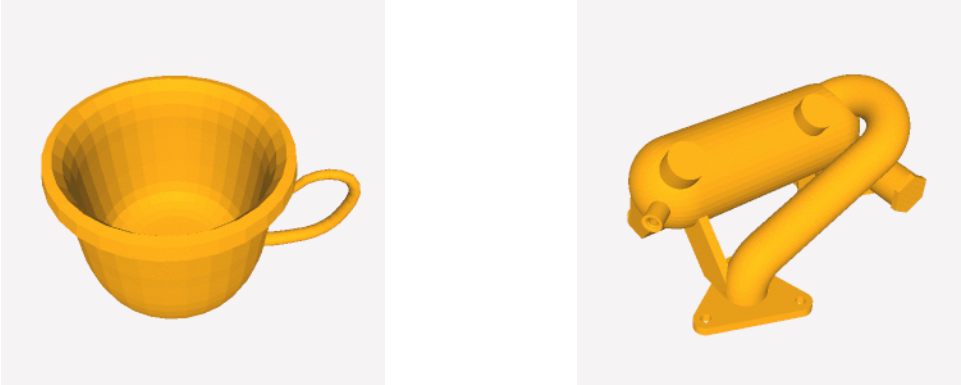


Figure 55: The two objects for which we tested projected area estimations. Left, a teacup, with 1000 polygons. Right, a pipe fitting, with 11,000 polygons.

4.5.1. Hierarchy Bounds Errors

For every cluster in the face cluster hierarchy of a model, a large number of directions distributed evenly over the directional sphere were sampled. For each direction, the actual (unoccluded) projected area, and the upper and lower bounds on the projected area were calculated. I then calculated the root-mean-square error of the bounds over all directions, as a fraction of the total surface area of the cluster. **Figure 56** shows the results plotted for both objects against the node depth. (That is, the depth of the node in the hierarchy.) As we would hope, the error over all directions decreases with depth for both objects. This trend is somewhat noisier in the case of the coarser cup model.

Figure 57 shows the results of plotting the error in the lower bound only (that due to our sum-area normal calculation.) This shows that our clusters are getting flatter and thus our bounds are getting tighter as we descend in the hierarchy. Of the lower and upper bounds, the lower seems to provide a better fit to the actual projected area, which provides some justification for only using the sum-area normal in the actual radiosity calculations. Note that the error shown here is the error between the unoccluded projected area and our lower bound; the error in the bound's representation of the visible projected area is at most this, and possibly smaller.

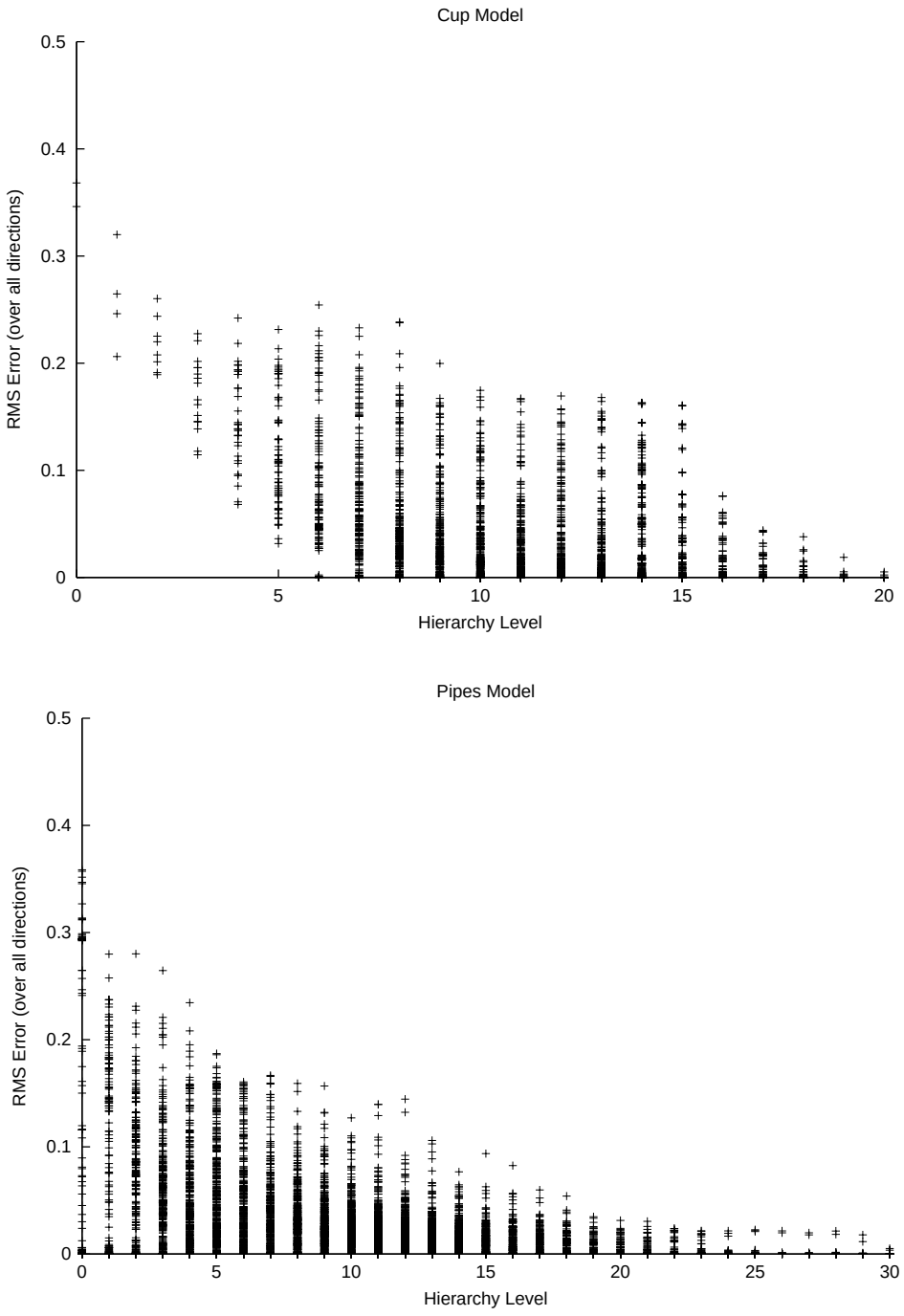


Figure 56: Performance of bounds over the entire face cluster hierarchy. The root cluster is at level zero, and the leaves are at the highest levels.

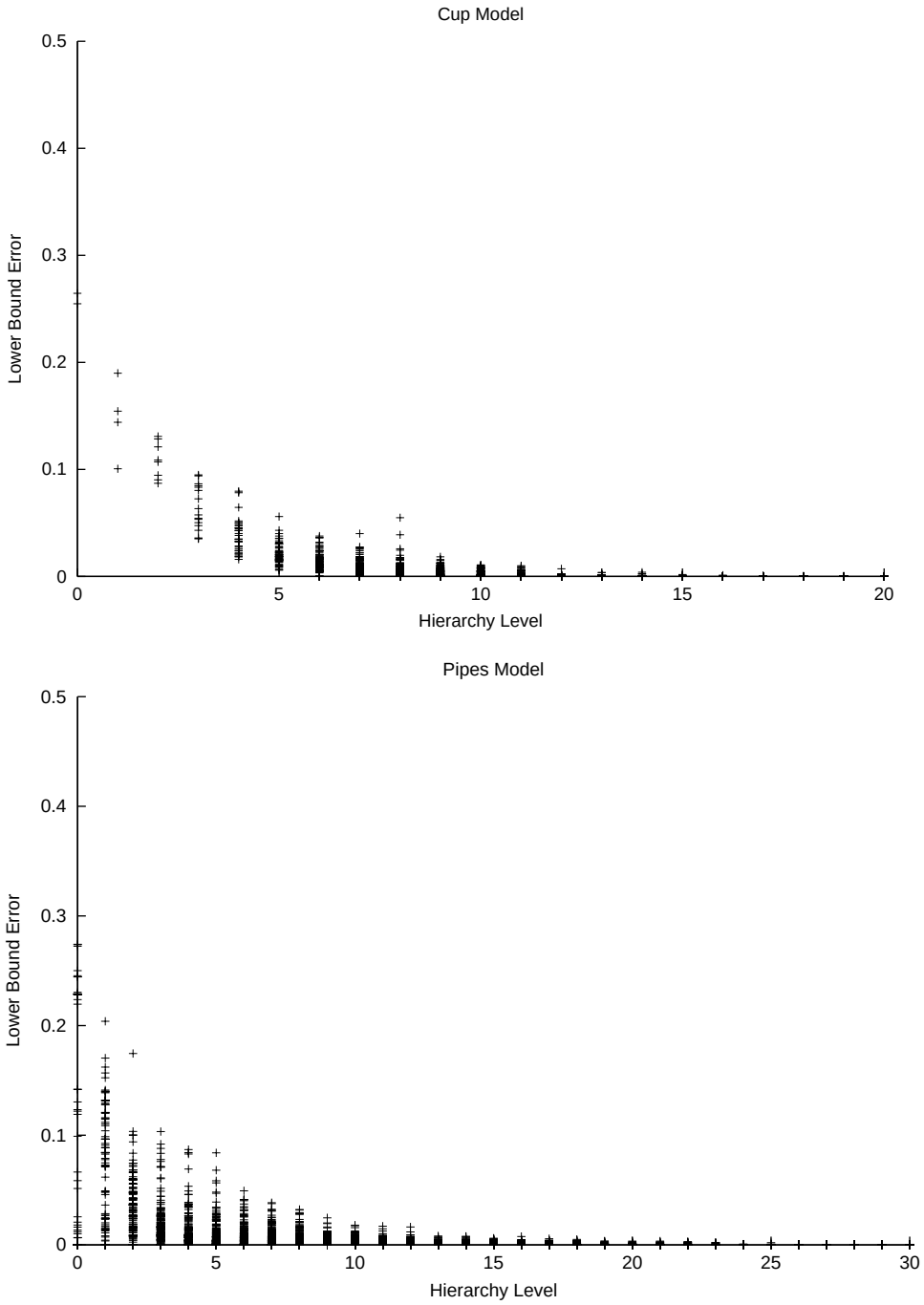


Figure 57: Error in the lower bound on the projected area. Top: for the cup mode the initially high relative error decreases with depth in the hierarchy. Bottom: for the pipes model, even at the highest levels of the hierarchy the lower bound for some clusters is exact.

4.5.2. Cluster Error Bounds

I also looked at the performance of the bounds over several individual cluster nodes, two from each model. **Figure 58** shows the relevant clusters from the cup model, and **Figure 59** shows the clusters from the pipes model. The bounds can be shown relative to the direction from the cluster by plotting them against c, the cosine of the angle between the direction and the "up" direction of the bounding box. Thus at $c = 1$ we are looking at the cluster from directly overhead, at $c = 0$, we are looking at the cluster from side-on, and at $c = -1$ from the rear. Again, we measure the error in the bounds as the difference between our upper and lower bounds, relative to the total surface area of the cluster.

Figure 58: Two clusters from the cup model. The cluster on the left corresponds to the bottom of the cup, and lies three levels deep in the hierarchy. The cluster on the right is the immediate parent of that cluster, and is considerably less planar in character.

Figure 60 shows the results for the cup clusters, and **Figure 61** the results for the pipes clusters. In both cases, while the error curve is large for the less directional clusters, for the smaller clusters it flattens out considerably. We would expect that, as we get deeper in the hierarchy, projected areas for $c \leq 0$ will go to zero, as the clusters get flatter.

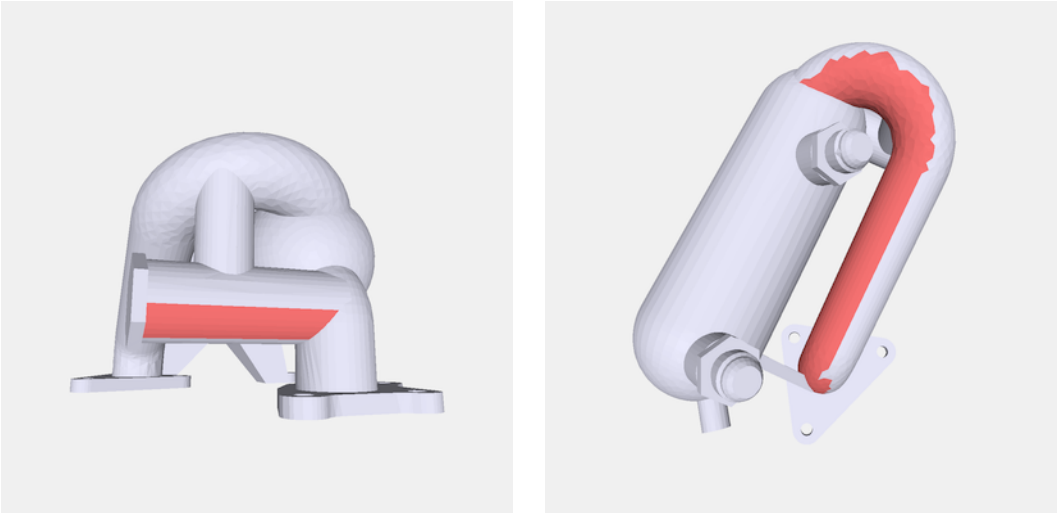


Figure 59: Two clusters from the pipes model. On the left, a smaller, flat cluster from level five of the hierarchy, and on the right, a larger cluster from level three.

Figure 62 shows how the upper and lower bounds behave as the cluster gets flatter. The lower bound tends to be accurate both immediately in front of and behind the cluster, and most inaccurate "side-on" to the cluster, around $c = 0$, where its assumption that the cluster is perfectly flat proves false. The upper bound, on the other hand, has greatest error outside this side-on region, although its error also goes to zero (though not nearly as quickly) at the front, side and back ($c = 1$, $c = 0$ and $c = -1$) of the cluster, because this is aligned with our VPA sample directions. Thus, luckily, the regions of error for both estimates cancel out to some extent.

4.5.3. Projected Area Functions

While graphing these results makes it easier to directly compare the various projected area estimates, for a more intuitive idea of what is happening, we can plot the various estimates in three dimensions as $PA(\hat{\mathbf{n}})$, that is, the projected area over the sphere of all directions. For the "cup bottom" cluster, **Figure 63** shows such plots for the actual projected area of the cluster, as well as our bounds for that area. The actual projected area function extends mainly downwards, with a

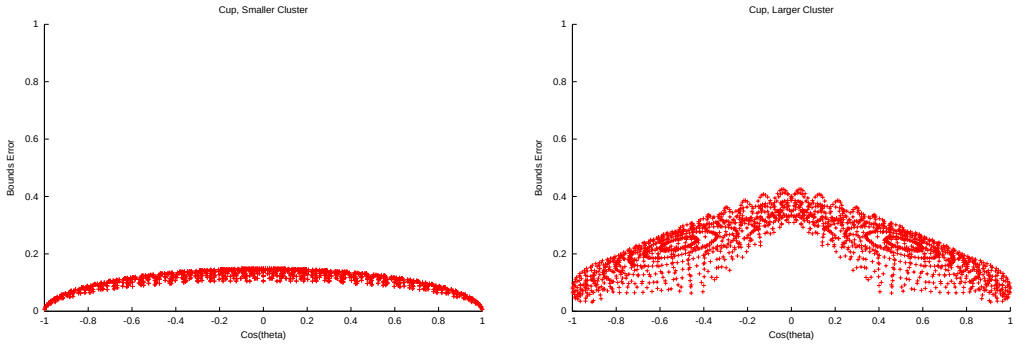


Figure 60: Error in the bounds for the cup clusters.

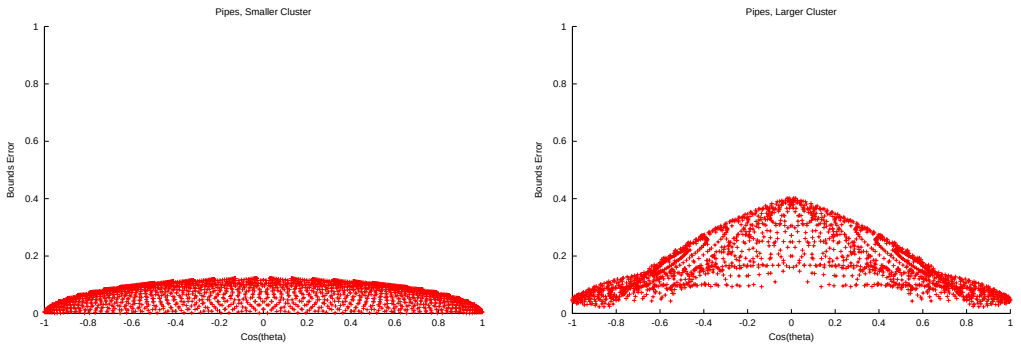


Figure 61: Error in the bounds for the pipes model's clusters.

little ring around the horizontal, and an indentation at the top where the projected area goes to zero as we move directly behind the surface. The lower bound is a simple cosine lobe in the direction of the cluster, while we can see that the upper bound consists of one large lobe in the direction of the cluster, with four much smaller lobes corresponding to the side areas of the cluster. The sixth lobe, corresponding to the sixth area sample, is non-existent, due to the zero projected area behind this cluster. The final image in the sequence shows how closely the two bounds enclose the actual projected area in this case.

Figure 64 shows the same results for the larger, decidedly non-planar cluster which covers part of the side of the cup as well. While the upper bound is much looser for this cluster, indicating the need for further subdivision, note that the lower bound still gives a surprisingly good fit to the function.

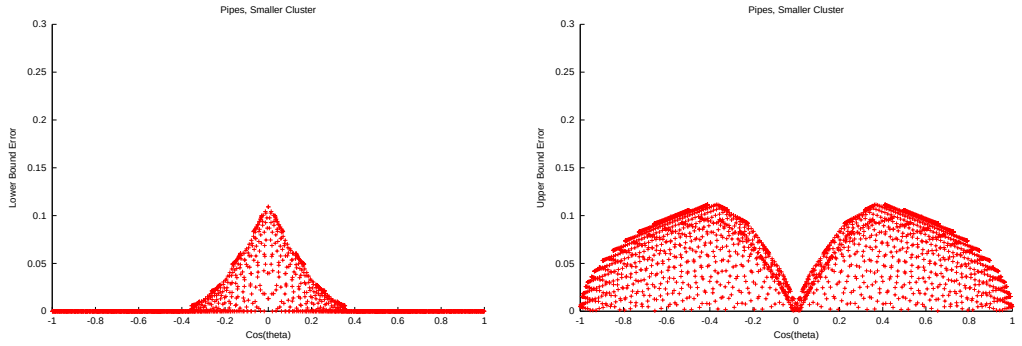


Figure 62: Upper and lower bound error for the small pipes cluster. The errors in the two bounds offset each other for largely flat clusters.

4.5.4. Upper Bound Strategies

I investigated how my preferred strategy of finding an upper bound for a cluster's projected area, by interpolating projected area samples from the axes of its OBB, compared to two other possible strategies. Firstly, we could take samples along the axes of an axis-aligned bounding box; this saves us having to transform the sample direction into the coordinate system of the OBB. Comparing to this strategy is also a good measure of how important picking "good" sample directions is. Secondly, we could follow the approach of previous researchers, and use the projection of the bounding box itself as an upper bound on the projected area of its contents [Stam97c].

Figure 65 compares these approaches for the two clusters from the pipes model. For the large cluster, both sampled approaches clearly outperform the projected bounding box approach, and the oriented sample approach is marginally better than the axis-aligned one. For the small, more planar cluster, the oriented sample strategy does much better than the axis-aligned one; picking good sample directions becomes more important for such a highly directional cluster. The projected bounding box approach deals particularly badly with angles "behind" the small cluster, because it takes no account of face orientation.

4.5.5. The SUPA as a lower bound on the VPA

There were a number of reasons given in section **Section 4.3.3** as to why the signed unoccluded projected area, SUPA, should only rarely overestimate the visible projected area, VPA. However, we would like to show some empirical sup-

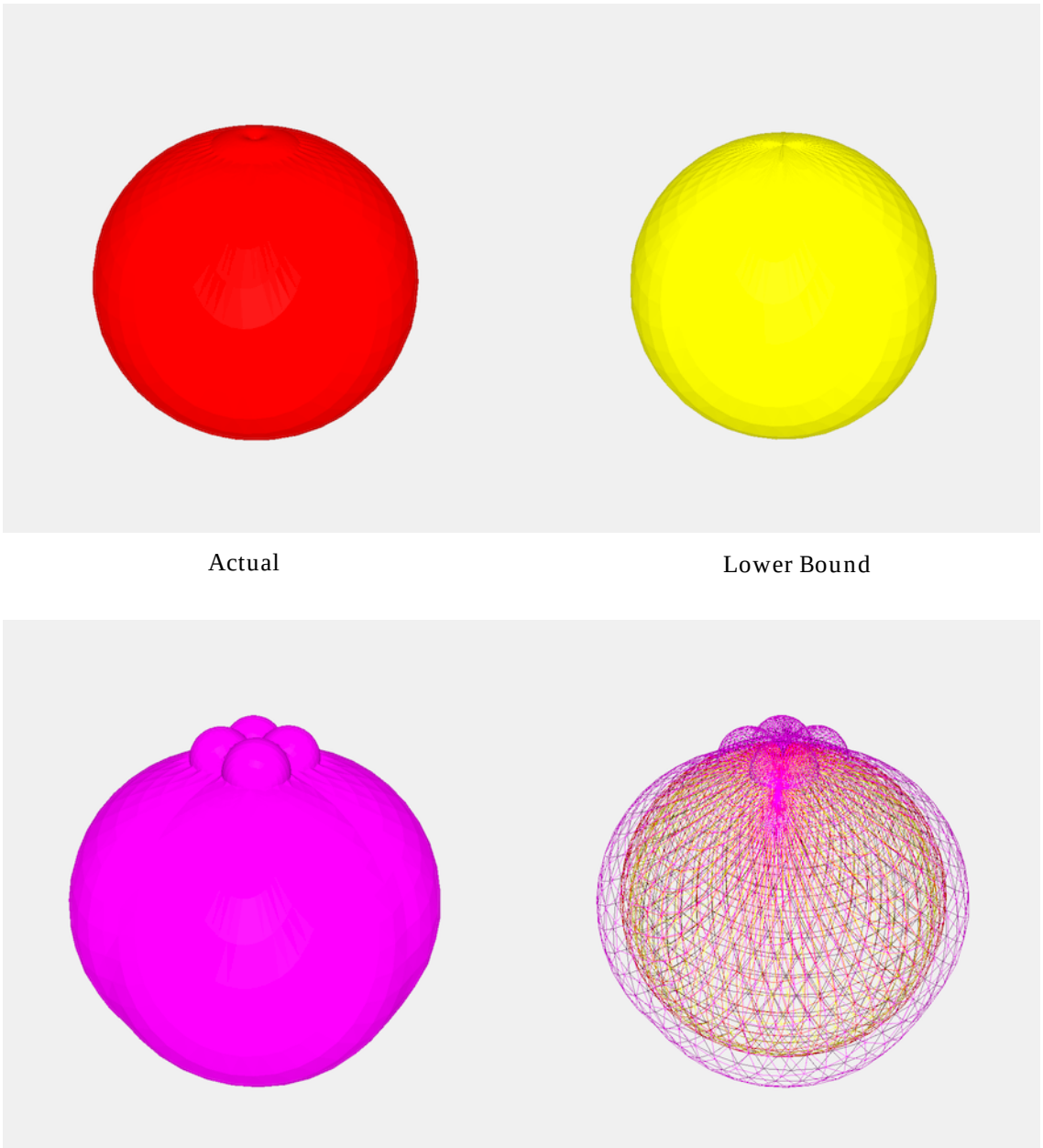
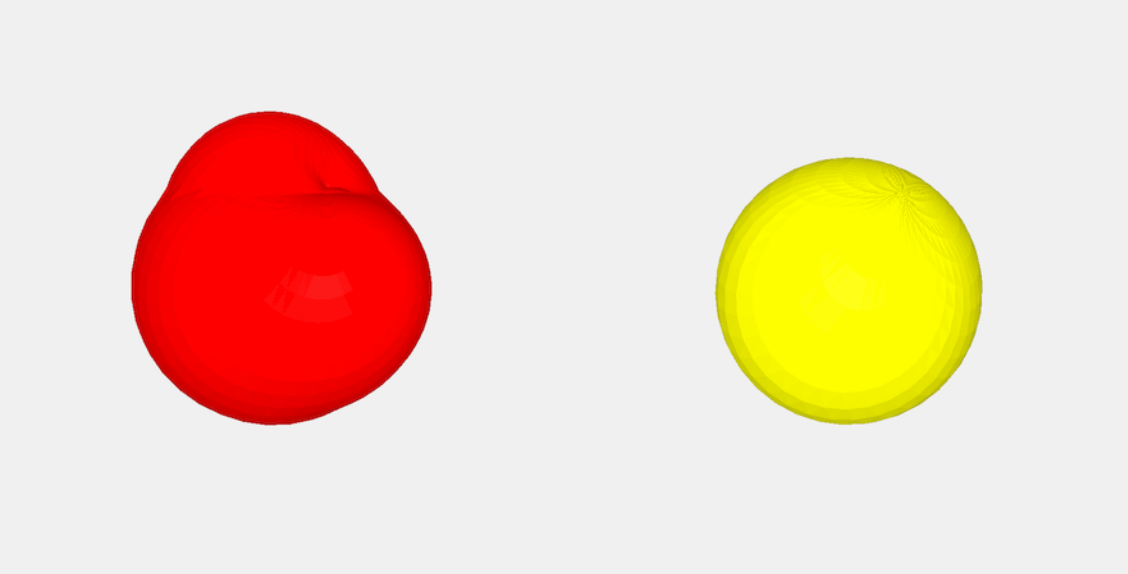
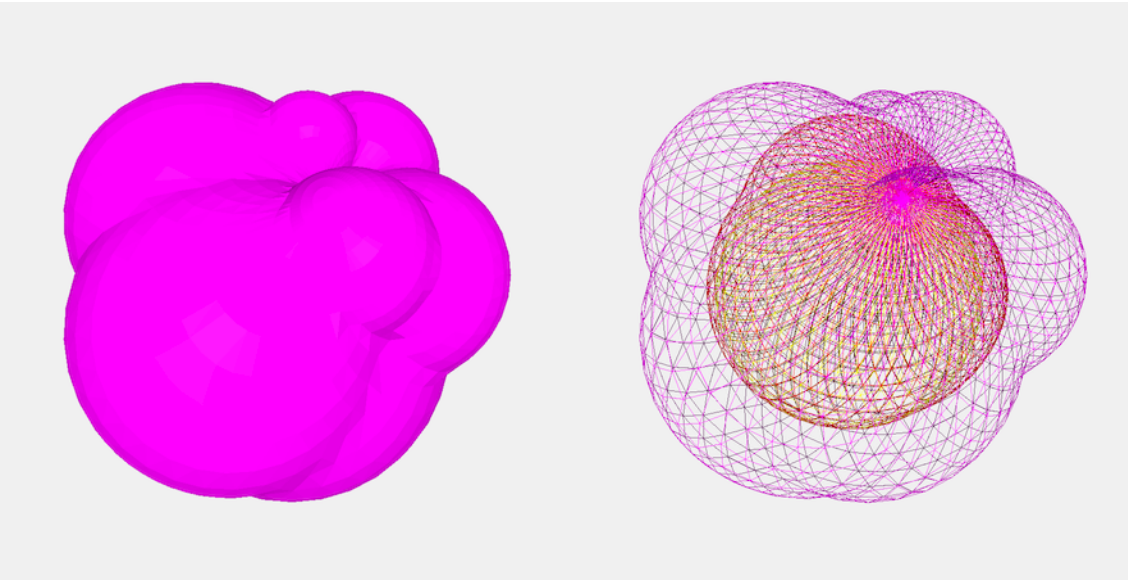


Figure 63: Projected area as a function of direction, cup bottom. From left to right we see the actual projected area of the cluster, the sum-area normal estimate of the area, and the sampled estimate. The last image shows all three functions together, illustrating how they nest correctly.



Actual

Lower Bound



Upper Bound

Nested

Figure 64: Projected area as a function of direction, cup side. The cup side approximates a flat surface less well, and hence the fit of the lower bound in particular is looser than that for the cup bottom.

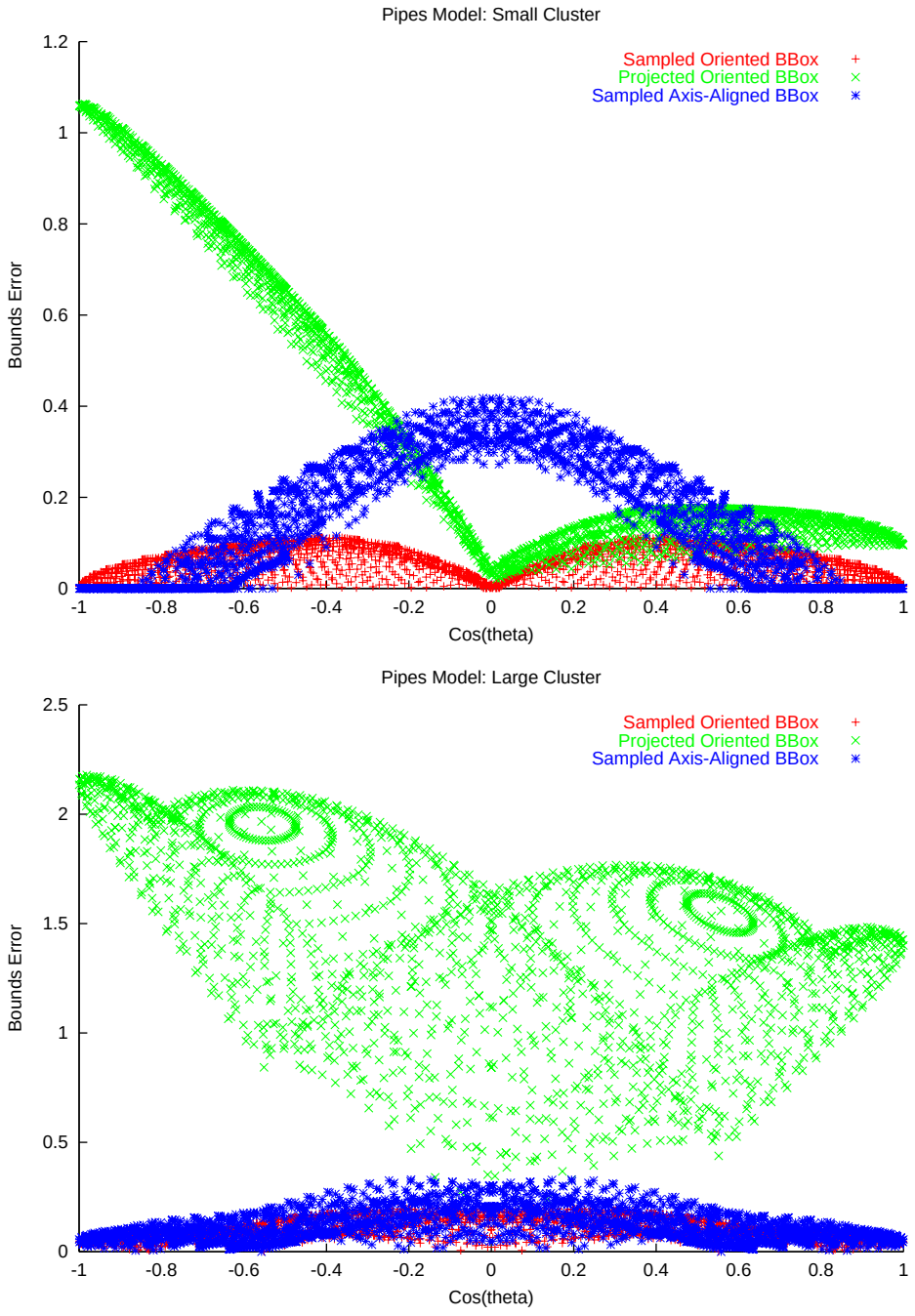


Figure 65: Different approaches to finding the upper bound. Top, errors for the small pipes cluster, and bottom, errors for the large pipes cluster. Using sampled projected areas along the axes of the oriented bounding box of the cluster gives better results than using samples on an axis-aligned bounding box, or the projected area of the oriented box.

port for these observations. To this end, we can test a number of models and their associated cluster hierarchies in the hope of finding the following:

- How often the SUPA fails to be a lower bound on the VPA. (We will refer to this as a bounds exception.)
- By how much it fails. Even if it only happens occasionally, if the SUPA is sometimes wildly wrong, its value as a lower bound estimate is much reduced. If it is close to the VPA even in situations where the bound is violated, however, its value is high.
- What kind of situations it fails in.

Unfortunately, accurately finding the visible projected area is more difficult than for the various other projected area metrics. We could employ analytic methods to calculate the silhouette of the cluster from a particular direction exactly. However, doing this for a large number of direction samples, and for potentially every cluster in a real model with a large number of faces, could take weeks of computation time. We must strike a balance between the accuracy of our VPA calculation, and the ability to test a large number of clusters for potential SUPA bounds violations.

I chose to use sampling to estimate the VPA, as follows. For each face in the cluster, a single ray was traced from the face's centre in direction $\mathbf{m}$. If the ray did not hit another face in that cluster, the face's projected area was added into the total. This approach is not ideal, as the resulting estimate of the VPA can underestimate the actual VPA, due to a face being counted as totally occluded when it is partially unoccluded. Thus when we test the SUPA against this estimate, there is a chance the SUPA will be larger than the VPA due to error in the estimate, rather than because it is actually larger than the VPA. Again, this is a compromise between total running time and accuracy in measuring errors in our lower bound; as it stands, each model test required the casting of over 1.5 billion rays, and a number of hours to run.

Using this technique I measured $\Delta x = (\mathrm{SUPA} - \mathrm{VPA})/A$, namely, the difference between the SUPA and VPA as a fraction of the cluster's total surface area, over a large number of directions[1], for every cluster in a number of models. Whenever this number was positive, a bounds exception was registered, and the direction and cluster involved were logged. Also, the maximum Δx for each cluster, $\Delta x_{\max}$, was logged.

The results for three models, each containing 100,000 polygons, are shown in **Table 4**. (The models can be seen in **Figure 79**.) Because some number of the

1. 484 directions distributed evenly over the sphere.

positive values of Δx will be occurring because of inaccuracies in the VPA estimate, and because we are more concerned with larger violations, exception counts are shown for a number of thresholds. Overall the results are as we might hope: a very small number of exceptions for each model, and none of these particularly large. The Buddha model suffers from the most exceptions, and the Isis model the least. This makes some sense, as the Buddha has a number of holes and areas of sharp curvature, whereas the Isis is reasonably smooth, and has no holes. No model has more than 0.2% exceptions where $\Delta x > 0.01$, that is, where the error is more than one hundredth of the cluster's surface area.

The average maximum exception is also shown, along with the largest such maximum. Clusters with less than 16 faces were ignored for the purposes of calculating the largest maximum, as clusters with so few faces suffer from considerable error in the VPA estimate. These largest maxima tend to be reasonable close to the average. Overall, we can conclude that bounds exceptions, when they do occur, are acceptably small.

Finally, one such bounds exception, from the Buddha model, is shown in **Figure 66**. Almost all exceptions examined where some error is obvious show this general pattern.

Model	**Percentage of Bounds Exceptions**			**Avg.** $\Delta x_{\max}$	**Max.** $\Delta x_{\max}$
	$\Delta x > 0.0$	$\Delta x > 0.001$	$\Delta x > 0.01$		
Buddha	0.52%	0.44%	0.20%	0.0043	0.066
Dragon	0.38%	0.32%	0.14%	0.0034	0.087
Isis	0.24%	0.18%	0.05%	0.0015	0.068

Table 4: Statistics for lower bound exceptions: SUPA > VPA.

4.6. Summary

In this chapter we have analysed the calculation of radiosity transfer between face clusters, and derived useful bounds and error estimates for our technique. Our bounds are almost always conservative (that is, not violated), and in practice we have found our bounds to be tighter than those previously published. Due to the visible projected area rule, using the sum-area normal vector **S** in our radiosity calculations can give surprisingly good results. Recapping:

- A good estimate of a cluster's *visible projected area*, the VPA, is critical to my algorithm.

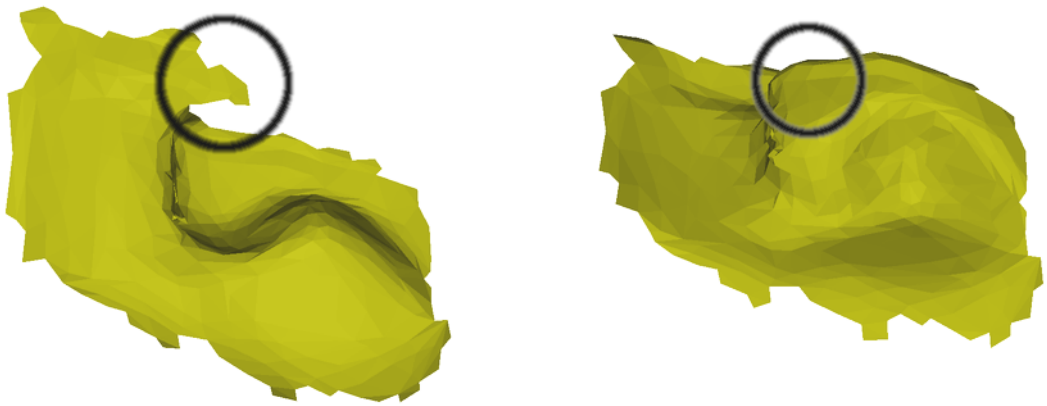


Figure 66: An example of one of the larger SUPA / VPA bounds violations from the Buddha model. The surface spur marked on the left is obscured in the rotated view shown on the right, but is still counted in the SUPA, leading to an exception. (An orthographic projection was used for both pictures.)

- $(\mathbf{m} \cdot \sum \mathbf{S})_+$, which I call the signed unoccluded projected area, or SUPA, is an approximate lower bound on the VPA.
- In my algorithm, this bound is rarely violated, because of the tendency of face clusters to be flat and well-shaped.
- The lower bound may only be violated when a cluster's projected boundary has regions with a winding number greater than one.
- As an upper bound, we can sample the non-negative unoccluded projected area, the NUPA, in six axial directions corresponding to the bounding box, and interpolate these samples. This bound is conservative.

Both bounds are constant time to evaluate in the number of polygons a cluster contains. They rely on preprocessing of a cluster, which

- for the lower bound is constant time to calculate from its children.

- for the upper bound is linear in the number of polygons in the cluster—no worse than the requirement for the calculation of bounding box extents.