RenderGen Notes

Nicolas Feltman, Yong He, Kayvon Fatahalian

November 4, 2012

1 Fragment Calculus

A fragment group is represented as a partial function from samples to geometry. Let $\oplus$ be a function which merges two fragment groups.

2 The Monophasic Renderer Model

This model takes in both the geometry and the samples at the same time and computes the results directly.

2.1 Utility Nodes

2.1.1 Fixed Points

$$\frac{\cdot}{\langle \texttt{fix}(N,l),\kappa,\Phi,G,S\rangle \to \langle N,\kappa,[\Phi,l:N],\Phi,G,S\rangle} \tag{1}$$

$$\frac{\cdot}{\langle \texttt{call}(l),\kappa,\Phi,G,S\rangle \to \langle \Phi(l),\kappa,\Phi,\Phi,G,S\rangle} \tag{2}$$

2.2 Decompositional Nodes

2.3 Filter Work Item

$$\frac{\gamma \between \sigma}{\texttt{Filt}(\gamma,\sigma,A) \to A} \tag{3}$$

$$\frac{\gamma \not\between \sigma}{\texttt{Filt}(\gamma,\sigma,A) \to \bot} \tag{4}$$

2.3.1 Identity Node

$$\frac{\cdot}{\langle \texttt{id}, \kappa, \Phi, G, S\rangle \to \langle \kappa, \emptyset, \Phi, G, S\rangle} \tag{5}$$

2.3.2 Full Decomposition Nodes

$$\frac{\cdot}{\texttt{Bag}(\dots, A, \dots, B, \dots) \to \texttt{Bag}(\dots, B, \dots, A, \dots)} \tag{6}$$

$$\frac{A \to A'}{\texttt{Bag}(A, \dots) \to \texttt{Bag}(A', \dots)} \tag{7}$$

$$\frac{\cdot}{\texttt{Bag}(\bot, A, \dots) \to \texttt{Bag}(A, \dots)} \tag{8}$$

$$\frac{\cdot}{\texttt{Bag}(\bot) \to \bot} \tag{9}$$

$$\frac{\cdot}{\texttt{Bag}(f_1, \dots, f_n) \to f_1 \oplus \cdots \oplus f_n} \tag{10}$$

First, `1S`:

$$\frac{S = \{s_1, \dots, s_n\}}{\langle \texttt{1S}, \kappa, \Phi, G, S\rangle \to \texttt{Bag}(\langle \kappa, \emptyset, \Phi, G, s_1\rangle, \dots, \langle \kappa, \emptyset, \Phi, G, s_n\rangle)} \tag{11}$$

And now `1G`:

$$\frac{G = \{g_1, \dots, g_n\}}{\langle \texttt{1S}, \kappa, \Phi, G, S\rangle \to \texttt{Bag}(\langle \kappa, \emptyset, \Phi, g_1, S\rangle, \dots, \langle \kappa, \emptyset, \Phi, g_n, S\rangle)} \tag{12}$$

2.3.3 Geometry Partitions

$$\frac{(G_1, G_2) = p(G) \quad \gamma_1 = \mathit{bound}(G_1) \quad \gamma_2 = \mathit{bound}(G_2)}{\langle \texttt{2\_GP}(p), \kappa, \Phi, G^{\gamma}, S^{\sigma}\rangle \to \texttt{Bag}(\langle \kappa, \emptyset, \Phi, G_1^{\gamma_1}, S^{\sigma}\rangle, \langle \kappa, \emptyset, \Phi, G_2^{\gamma_2}, S^{\sigma}\rangle)} \tag{13}$$

2.3.4 Samples Partitions (TODO)

I next present the basic sample partition.

$$\frac{}{\langle \texttt{2\_GP}(p, \gamma), \kappa, \Phi, G\rangle \to \texttt{Bag}(G)} \tag{14}$$

$$\frac{\{t_i\} = \mathit{split}(t)}{\langle \texttt{16}^2\texttt{\_SP}(G), S, t\rangle \to \langle \lambda(F). \bigcup F, \{(G, b, S_i, t_i) : b \between t_i\}\rangle} \tag{15}$$

2.4 Modifier and Combinator Nodes

The following nodes do not directly represent decomposition, but instead denote ways to combine or modify the basic nodes.

2.4.1 Branching

$$\frac{p(G,S) = \text{left}}{\langle N +_p M, \kappa, \Phi, G^\gamma, S^\sigma\rangle \rightarrow \langle N, \kappa, \Phi, G^\gamma, S^\sigma\rangle} \tag{16}$$

$$\frac{p(G,S) = \text{right}}{\langle N +_p M, \kappa, \Phi, G^\gamma, S^\sigma\rangle \rightarrow \langle M, \kappa, \Phi, G^\gamma, S^\sigma\rangle} \tag{17}$$

2.4.2 Composition (TODO)

The composition of decompositions is denoted with a >=>.

$$\frac{.}{\langle N{>=>}M, \emptyset, \Phi, G^\gamma, S^\sigma\rangle \rightarrow \langle N, M, \Phi, G^\gamma, S^\sigma\rangle} \tag{18}$$

$$\frac{\kappa \neq \emptyset}{\langle N{>=>}M, \kappa, \Phi, G^\gamma, S^\sigma\rangle \rightarrow \langle N, M{>=>}\kappa, \Phi, G^\gamma, S^\sigma\rangle} \tag{19}$$

2.4.3 Temporary Decompositions

forthcoming

3 The Biphasic Renderer Model (ALL TODO)

A renderer is entirely specified by a tree of source nodes. The renderer operates in two phases:

- **Precompute Phase:** This phase takes the source nodes and a set of geometry and produces a structure which holds all of the geometry.
- **Evaluation Phase:** The input to the evaluation phase is the samples, as well as the structure that was build by the precomput phase. (Lazy BVH build here)

3.1 Node Summary

(NOTE: ignore the content of these tables; they are not being synced)

At a glance, here are all of the source nodes.

Name	Node Arity	Special Arguments	Notes
`id`	0	-	-
`16`^{2}`_SP`	0	σ	-
`2_GP`	0	p, γ	γ is a bound type argument
`1S`	0	-	-
>=>	2	-	-
`Plus`	2	p, m	mode m determine's p's input
`fix`	1	l	-

Here is a table of all of the geometry structure elements. Obviously, elements like `Left` and `Right` could be combined into a single node with a binary special argument, if one preferred that representation.

Name	Node Arity	Special Arguments	Notes
`Bag`	0	-	-
`OpenNode`	2	$b_1 : \gamma$, $b_2 : \gamma$	-
`Left`	1	-	-
`Right`	1	-	-
`Plus`	2	-	-

We also have some sample structure elements.

Name	Node Arity	Special Arguments	Notes
`Bag`	0	S	-

The input to the first phase

3.2 Utility Nodes

3.2.1 Fixed Points

Recursion requires two nodes: one to set a fixed-point label, and another to call it. The nodes are called `fix` and `call`, respectively. The former is node-unary, the latter is node-nullary, and both require a single special argument, which is the name of the label in use. Note that a label introduced at a `fix` node is only in scope in the subtree beneath that node. Also, when drawing the tree graphically, $\texttt{call}(l)$ is usually represented with just the label l.

Semantically, this behavior is handled by keeping about an environment variable Φ, which maps labels to nodes. The environment is untouched by all nodes except `fix` and `call`. In particular, `fix` introduces labels to the environment, and `call` reads labels from the environment. Noting that $[\Phi, l : N]$ denotes the function Φ extended with the output N at input l, we have:

$$\frac{\cdot}{\langle \texttt{fix}(N, l), \kappa, \Phi, G\rangle \rightarrow \langle N, \kappa, [\Phi, l : N], G\rangle} \tag{20}$$

$$\frac{\cdot}{\langle \texttt{call}(l), \kappa, \Phi, G\rangle \rightarrow \langle \Phi(l), \kappa, \Phi, G\rangle} \tag{21}$$

Then during the evaluation phase, we have,

$$\frac{\langle N, [\Phi, l : N], R, S\rangle \rightarrow \langle m, F\rangle}{\langle \texttt{fix}(N, l), \Phi, R, S\rangle \rightarrow \langle m, F\rangle} \tag{22}$$

$$\frac{\langle \Phi(l), \Phi, R, S\rangle \rightarrow \langle m, F\rangle}{\langle \texttt{call}(l), \Phi, R, S\rangle \rightarrow \langle m, F\rangle} \tag{23}$$

3.3 Decompositional Nodes

In this subsection, I'll cover all of the nodes which perform decompositions. All of these nodes are nullary, meaning that they do not take any arguments.

3.3.1 Identity Node

The identity node, noted id, represents the trivial decomposition.

$$\frac{\cdot}{\langle \mathtt{id}, \kappa, \Phi, G\rangle \to \langle \kappa, \emptyset, \Phi, G\rangle} \text{ (Precompute Phase)} \tag{24}$$

$$\frac{\cdot}{\langle \mathtt{id}, R, S\rangle \to \langle \cdot, [(R,S)]\rangle} \text{ (Eval Phase)} \tag{25}$$

3.3.2 Full Decomposition Nodes

I next note the full decomposition nodes, 1S and 1G. These decompositions completely decompose their respective sets. First we cover 1S:

$$\frac{\cdot}{\langle \mathtt{1S}, \kappa, \Phi, G\rangle \to \langle \kappa, \emptyset, \Phi, G\rangle} \tag{26}$$

$$\frac{\cdot}{\langle \mathtt{1S}, R, S\rangle \to \langle \lambda(F).\bigcup F, \{(R,s) : s \in S\}\rangle} \tag{27}$$

And now 1G:

$$\frac{\cdot}{\langle \mathtt{1G}, \emptyset, \Phi, G\rangle \to \mathtt{Bag}(G)} \tag{28}$$

$$\frac{\cdot}{\langle \mathtt{1G}, \mathtt{Bag}(G), S\rangle \to \langle \ldots, \{(g,S) : g \in G\}\rangle} \tag{29}$$

3.3.3 Geometry Partitions

I next present the basic geometry partition. In this case, the node partitions into two sets, but the generalization to more is obvious. The parameter γ identifies a module for producing bounds (for examples, an axis aligned bounding box, or half-space bounds, etc.).

$$\frac{(G_1, G_2) = p(G) \quad b_1 = \mathit{bound}_\gamma(G_1) \quad b_2 = \mathit{bound}_\gamma(G_2)}{\langle \mathtt{2\_GP}(p,\gamma), \kappa, \Phi, G\rangle \to \mathtt{OpenNode}(\langle \kappa, \emptyset, \Phi, G_1\rangle, \langle \kappa, \emptyset, \Phi, G_2\rangle, b_1, b_2)} \tag{30}$$

$$\frac{A \to A'}{\mathtt{OpenNode}(A, B, b_1, b_2) \to \mathtt{OpenNode}(A', B, b_1, b_2)} \tag{31}$$

$$\frac{B \to B'}{\mathtt{OpenNode}(A, B, b_1, b_2) \to \mathtt{OpenNode}(A, B', b_1, b_2)} \tag{32}$$

The resulting OpenNode structure contains the bounds of the two children, and pointers to substructures (*i.e.*, one node in a BVH tree). This kind of node assumes that its bounds have been checked by the parent structure, or that bounds checking is not important.

In the following nodes, the symbol $\between$ means that two bounds intersect, and $\not\between$ means they do not intersect.

$$\frac{b_1 \not\between t \quad b_2 \not\between t}{\langle \texttt{2\_GP}(p,\gamma), \texttt{OpenNode}(n_1, n_2, b_1, b_2), S, t\rangle \to \langle \{\}, \{\}\rangle} \tag{33}$$

$$\frac{b_1 \between t \quad b_2 \not\between t \quad \langle n_1, S, t\rangle \to \langle f, P\rangle}{\langle \texttt{2\_GP}(p,\gamma), \texttt{OpenNode}(n_1, n_2, b_1, b_2), S, t\rangle \to \langle f, P\rangle} \tag{34}$$

$$\frac{b_1 \not\between t \quad b_2 \between t \quad \langle n_2, S, t\rangle \to \langle f, P\rangle}{\langle \texttt{2\_GP}(p,\gamma), \texttt{OpenNode}(n_1, n_2, b_1, b_2), S, t\rangle \to \langle f, P\rangle} \tag{35}$$

TODO: figure out how to carry around sample metadata

3.3.4 Samples Partitions

I next present the basic sample partition.

$$\frac{}{\langle \texttt{2\_GP}(p,\gamma), \kappa, \Phi, G\rangle \to \texttt{Bag}(G)} \tag{36}$$

$$\frac{\{t_i\} = \mathit{split}(t)}{\langle \texttt{16}^2\texttt{\_SP}(G), S, t\rangle \to \langle \lambda(F). \bigcup F, \{(G, b, S_i, t_i) : b \between t_i\}\rangle} \tag{37}$$

3.4 Modifier and Combinator Nodes

The following nodes do not directly represent decomposition, but instead denote ways to combine or modify the basic nodes.

3.4.1 Branching

Sometimes, the choice of what decomposition you want to perform is dependent on data. To express this situation, we have the branch operator. The expression $N\texttt{Plus}_p^m M$ denotes a conditional choice between the nodes N and M, based on predicate p, evaluating under one of three modes m (geometry, sample-duplicate, and sample-defer). The particular mode m determines the type of p. The first mode is for when the branch choice depends only on geometry, meaning that the decision can be made during the build phase. Note the rightmost judgement above the bar means "the node N, for samples G, will produce a constructor f to operate on the results of the breakdown C".

$$\frac{m = \text{geom} \quad p(G) = \text{left}}{\langle N +_p^m M, \kappa, \Phi, G\rangle \to \texttt{Left}(\langle N, \kappa, \Phi, G\rangle)} \tag{38}$$

$$\frac{m = \text{geom} \quad p(G) = \text{right}}{\langle N +_p^m M, \kappa, \Phi, G\rangle \to \texttt{Right}(\langle M, \kappa, \Phi, G\rangle)} \tag{39}$$

$$\frac{A \to A'}{\texttt{Left}(A) \to \texttt{Left}(A')} \tag{40}$$

$$\frac{A \to A'}{\texttt{Right}(A) \to \texttt{Right}(A')} \tag{41}$$

Next there is the case where the branch decision depends on samples and no action is taken during the build phase, probably because the user doesn't want to risk doing unnecessary decompositions.

$$\frac{m = \text{sDef}}{\langle N +_p^m M, \kappa, \Phi, G\rangle \to \texttt{Bag}(G)} \tag{42}$$

Alternatively, the user may want to perform both sub-decompositions proactively. This may be wasted work and wasted storage if no samples end up requiring the sub-expressions to be evaluated. However, if it's likely that both will end up evaluated, we'd like to perform the appropriate precomputation. The best example of this so far is Warren's frustrum rasterizer. In this case, the duplicate represntation down both sides of the tree has little memory overhead because a BVH subtree can be represented as a partition of an array.

$$\frac{m = \text{sDup}}{\langle N +_p^m M, \kappa, \Phi, G\rangle \to \texttt{Plus}(\langle N, \kappa, \Phi, G\rangle, \langle M, \kappa, \Phi, G\rangle)} \tag{43}$$

$$\frac{A \to A'}{\texttt{Plus}(A, B) \to \texttt{Plus}(A', B)} \tag{44}$$

$$\frac{B \to B'}{\texttt{Plus}(A, B) \to \texttt{Plus}(A, B')} \tag{45}$$

3.4.2 Composition

The composition of decompositions is denoted with a >=>. Note that previously, I have used variables like C_N to denote lists of sets of geometry that are output from the N decomposition. Below, I instead deconstruct C_N into the list $[G_1, \ldots, G_n]$, because I need to pass each G_i independently to the right-hand-side M decomposition.

$$\frac{\cdot}{\langle N{>=>}M, \emptyset, \Phi, G\rangle \to \langle N, M, \Phi, G\rangle} \tag{46}$$

$$\frac{\kappa \neq \emptyset}{\langle N{>=>}M, \kappa, \Phi, G\rangle \to \langle N, M{>=>}\kappa, \Phi, G\rangle} \tag{47}$$

3.4.3 Temporary Decompositions

forthcoming

3.4.4 Phase-Adjustment Node

forthcoming

4 Example Renderers

4.1 Basic BVH Ray-Tracer