HOUDINI FOUNDATIONS

都市構築用タスクワークフローの作成

パイプラインのワークフローの管理に Task Operator (TOP) を利用することができます。TOPは PDG (Procedural Dependency Graph) と呼ばれるテクノロジを使って構築されています。PDG に基づいて作成されたワークフローで使用される TOP ノードは、ローカルコンピュータまたは大規模なコンピュートファームにタスクを分散させるワークアイテムを生成します。

TOP ネットワークでは、各ワークアイテムが他のワークアイテムにどのように依存しているかや、各ワークアイテムが最終出力にどのように貢献するかを確認することができます。この情報はノードグラフとして容易に視覚化できます。ノードグラフを使えば、ネットワーク内でのデータフローを定義できます。TOP を使えば、パイプラインの自動化、分析、およびスケールに使用可能なワークフローをビルドすることができます。

このレッスンでは、TOP ノードを使って都市マップを取得し、都市区画ごとにビルを作成した後、このシステムを拡張してより複雑なビルや大規模な都市マップを処理できるようにします。Houdini のアーティストであればおそらく、SOP でこれを行う方法を知っていると思いますが、TOP を使えば、PDG のワークフローを学びながら、並列処理の対象となる複数のタスクを外部のコンピュートファームに分散させるように容易にスケール可能なシステムを作成することができます。

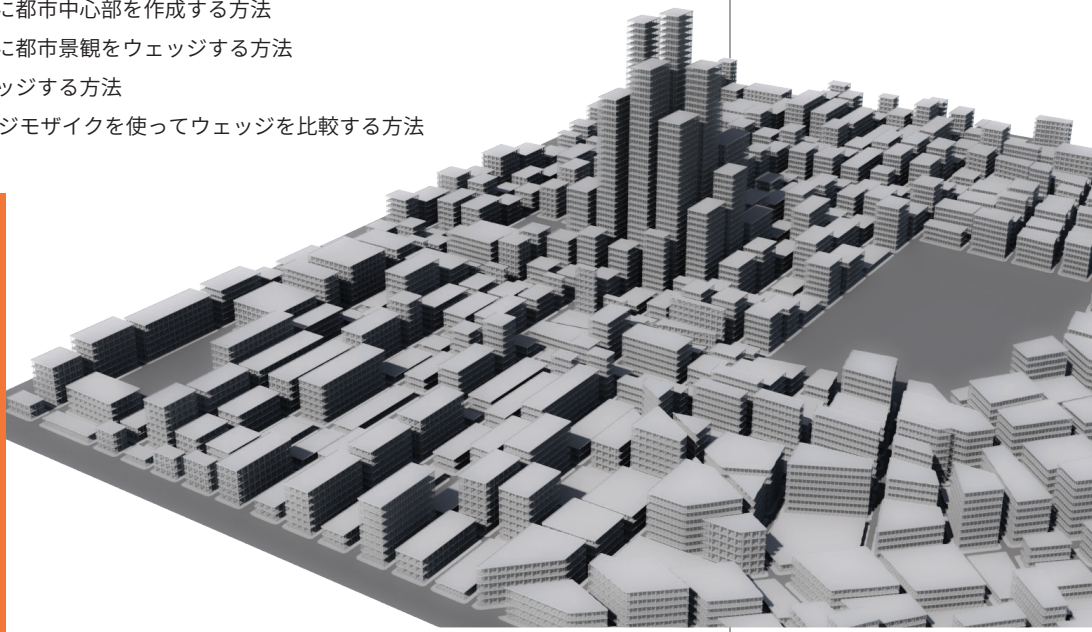
注: このレッスンでは Image Magick を使用します。このアプリケーションがコンピュータにインストールされていることを確認してください。

レッスンの目標

- プロシージャルな都市を構築した後、それをレンダーするような TOP（Task Operator）ネットワークを作成すること

学習内容

- 都市マップイメージをジオメトリに変換する方法
- 都市区画を保存する TOP ネットワークのセットアップ方法
- 各都市区画上にビルを構築するために TOP でジオメトリを作成する方法
- 一部のビルを他のビルより高くするために都市中心部を作成する方法
- 都市中心部のさまざまな位置を試すために都市景観をウェッジする方法
- さまざまなマップイメージの使用をウェッジする方法
- TOP を使って都市をレンダーし、イメージモザイクを使ってウェッジを比較する方法



レッスンの互換性

Houdini 17.5 の機能向けに執筆しています

このレッスンの手順は以下の Houdini 製品を使って実行できます。

Houdini Core	✔
Houdini FX	✔
Houdini Indie	✔
Houdini Apprentice	✔
Houdini Education	✔

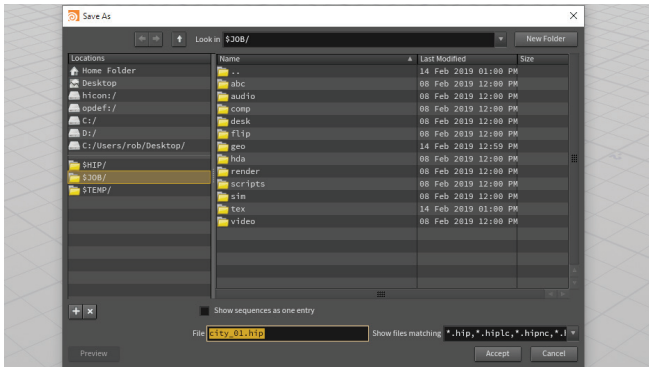
ドキュメントバージョン 1.0 | 2019 年 3 月

パート 1:
都市グリッドを作成する

プロシージャルな都市を構築するには、まず都市のグリッドを作成します。都市マップの白黒イメージを含むイメージファイルをトレースすることで、ジオメトリを作成します。これが、TOP で構築するネットワークの入力ジオメトリとなります。

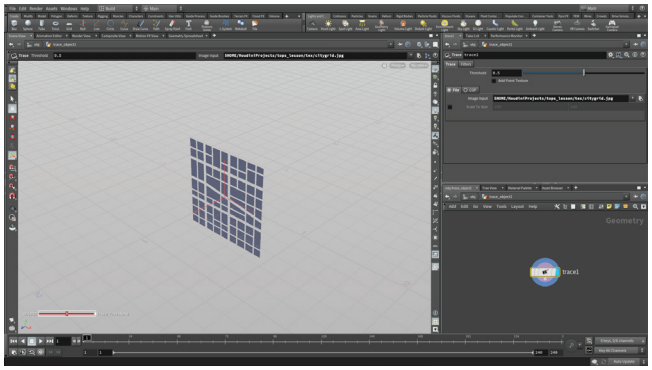
プロジェクトファイル

SideFX.com の**基礎チュートリアル**のページ（このチュートリアルを入手したページ）にアクセスし、*tops_lesson_start* ディレクトリをダウンロードします。それを、**home**ディレクトリまたは **documents** ディレクトリの中に存在している **Houdini Projects** ディレクトリ内に配置します。



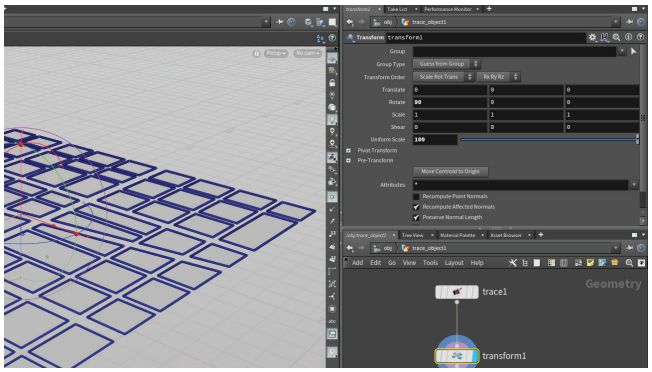
01 **[File] > [Set Project]** を選択します。事前にダウンロードしておいた *tops_lesson_start* ディレクトリを見つけ、**[Accept]** をクリックします。すると、このプロジェクトディレクトリとサブフォルダが、このショットに関連するすべてのファイルの格納場所として設定されます。

[File] > [Save As...] を選択します。新しく作成した *tops_lesson_start* ディレクトリが参照先になっているはずです。そうなっていない場合は、左側の列にある$JOBをクリックします。ファイル名を*city_01.hip* に設定し、**[Accept]** をクリックして保存します。



02 ビューポートで**タブキーを押して**メニューを開き、**TRACE** と入力し始めます。**[Trace]** を選択すると、シーン内への配置待ちの状態にあることを示す円の輪郭が、カーソルの位置に表示されます。**Enter キーを押して**原点の位置に配置します。

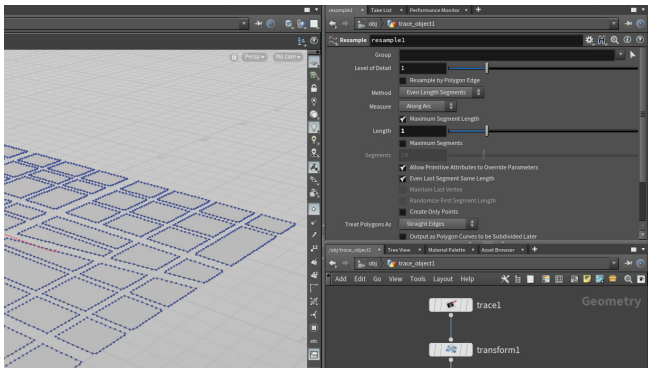
ネットワークビューで、ノードを**ダブルクリック**してジオメトリレベルに下がります。*trace* ノードを選択し、**[Image Input]** パラメータの横にある**ファイル選択ダイアログ**をクリックします。**$HIP** をクリックした後、*tex* ディレクトリ内に移動します。*citygrid.jpg* イメージを選択した後、**[Accept]** をクリックします。次に、**[Scale to Size]** オプションを**オン**にし、その値を **500, 500** に設定します。これにより精度が向上します。



03 ネットワークビューで、*trace* ノードの出力上で**右マウスボタンをクリック**し、**TRANSFORM** と入力し始めます。**Transform** ノードを選択し、クリックして *trace* ノードの下に配置します。その表示フラグを設定し、このノードの結果が表示されるようにします。以下の設定を行います。

- **[Rotate X]: -90**
- **[Uniform Scale]: 100**

transform ノードの直後に **Reverse** ノードを追加し、法線が上に向くようにします。ビューポートで、**スペース-H** キーを押して都市グリッドの全体を表示させます。**表示オプション**バーの **[Display Points]** をオンにすると、各都市区画の周りにトレースポイントが多数存在していることがわかります。



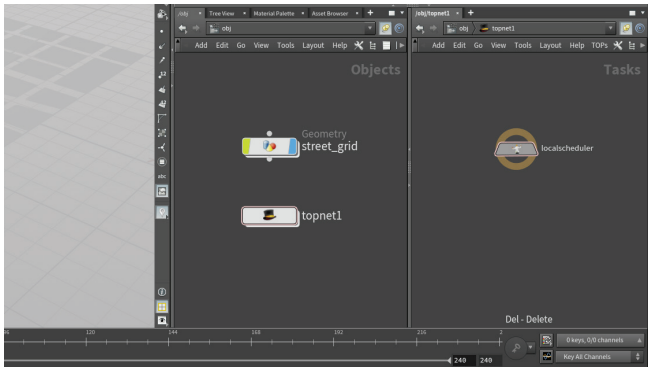
04 ネットワークビューで**タブキー**を押し、**RESAMPLE** と入力し始めます。**Resample** ノードを選択し、クリックして *transform* ノードの下に配置します。まだ接続はしないでください。**[Length]** を **1** に設定します。その後、*transform* ノードの出力を *resample* ノードの入力に接続します。

resample ノードの出力で**右マウスボタンをクリック**し、**[Null]** を選択します。クリックしてヌルノードをチェーンの末尾に配置します。その表示フラグを設定し、名前を *CITYBLOCKS_OUT* に変更します。

オブジェクトレベルに戻り、オブジェクトの名前を *street_grid* に変更します。作業内容を保存（**[Save]**）します。

パート 2:
ワークアイテムを生成および表示する

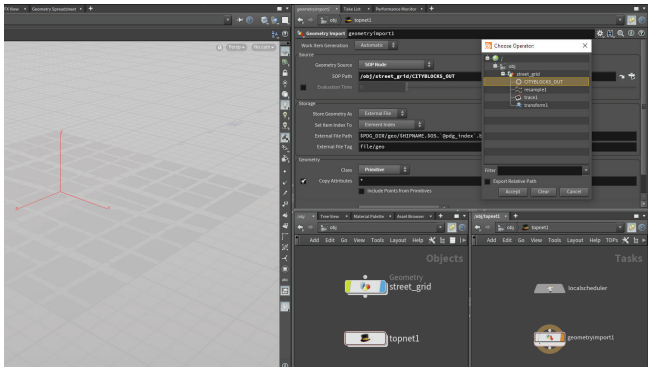
都市のグリッドが手に入ったので、異なる都市区画を個別のワークアイテムに分割できます。そうすれば、各区画に基づいてビルを生成できるようになります。各区画を保存するための TOP ネットワークをセットアップします。同時に、選択されたワークアイテムを視覚化し、望みどおりの結果が得られているか確認できるようにするためのオブジェクトも作成します。



01 オブジェクトレベルでネットワークビューに移動し、**タブキーを押して TOP...** と入力し始め、**[TOP Network]** を選択します。クリックしてネットワーク内にノードを配置します。

ネットワークビューの右上にある小さな**矢印**をクリックします。メニューから **[Split Pane Left/Right]** を選択します。左側のネットワークビューで **[Pin]** アイコンをクリックします。右側のネットワークビューはすでに固定されています。

右側のネットワークビューで、*topnet* を**ダブルクリック**してその内側に入ります。これで、都市を開発する際に両方のネットワークを操作できるようになりました。



02 TOP ネットワーク内で、**タブキーを押して [Geometry Import]** を選択します。クリックしてノードを配置します。**[Geometry Source]** を **[SOP Node]** に設定した後、**[SOP Path]** の横にある **[Choose Operator]** アイコンをクリックします。フローティングウィンドウを使って、*CITYBLOCKS_OUT* ヌルノードに移動して選択します。

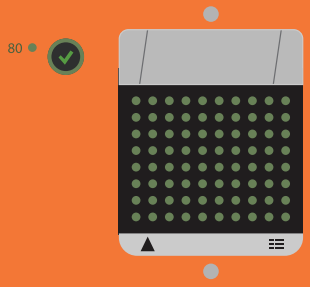
[Store Geometry As] は、**[External File]** に設定されたままにします。**[Class]** を **[Primitive]** に設定します。すると、ファイルがディスクに格納されるようになるので、次の TOP ノードからそれらのファイルを取得できます。これは特に、コンピュートファームにタスクを分散させる TOP ネットワークをセットアップする場合に重要となります。

03 この TOP ノードで**右マウスボタンをクリック**し、メニューから **[Cook Selected Node]** を選択します。あるいはこのノードを選択し、**Shift-G** キーを押してクックしてもかまいません。ワークアイテムを表す小さなドットが表示されます。これらは、ノードがクックされるにつれてタスクになります。

ドット上で Ctrl キーを押しながら中マウスボタンをクリックすると、ワークアイテムのアトリビュートが表示されます。アトリビュートには、**Index** のように、すべてのワークアイテムが備える基本的な TOP アトリビュートもあれば、ワークアイテムのタイプに固有のものもあります。各ワークアイテムは 1 つの出力ファイル（**Output**）に関連付けられていることがわかります。ワークアイテム上で**右マウスボタンをクリック**して **[View Task Output]** を選択すれば、格納されたジオメトリを個別のジオメトリビューアで確認することができます。

ワークアイテム

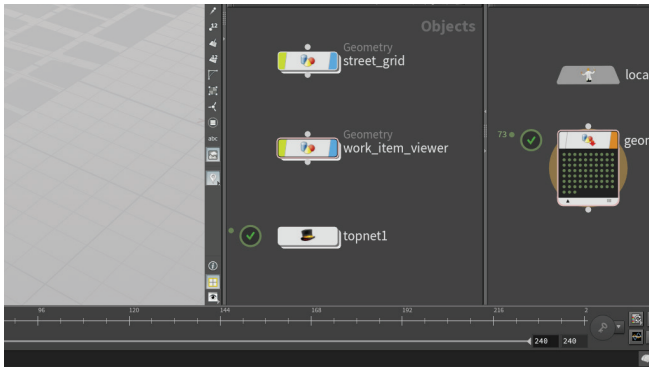
TOPノードは、実行を依頼されたタスクごとにワークアイテムを作成します。これらはTOPノード上でドットとして表現されます。これらのドットを使って各ワークアイテムのステータスを視覚化したり、特定のドットを選択してそのワークアイテムの進行状況を評価したりすることができます。すべてのワークアイテムのクックが完全に終了すると、ノードにチェックマークが付き、終了したワークアイテムが緑色になります。



スケジューラノードとは

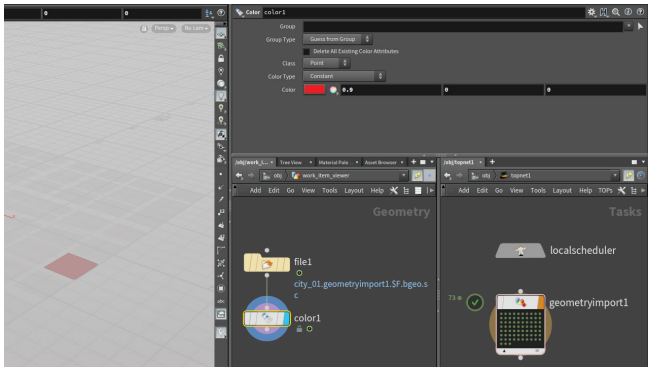
最初にTOPネットワークを作成する際には、クック対象のタスクを管理できるように、ローカルスケジューラノードが作成されます。ローカルスケジューラはローカルコンピュータを指しており、利用可能なコアの一部を使ってクックを行います。**[Maximum CPUs to Use]** オプションを使えば、その個数を利用可能なコアに合わせて設定できます。

また、より大規模なコンピュートファームにタスクを送信できるよう、**HQueue、Deadline、Tractor、Python** 用のスケジューラノードをセットアップすることもできます。この場合、並列に処理できるワークアイテムの数が増えるので、グラフの効率も向上します。



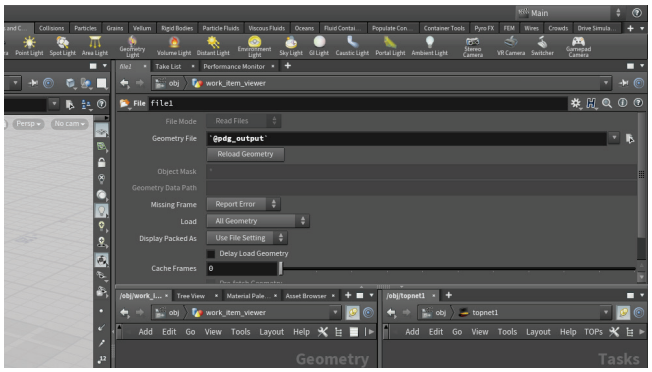
04 ワークアイテムをビューポートに表示させるには、ワークアイテムビューアをセットアップする必要があります。左側のネットワークビューで、**タブキーを押して [File]** を選択します。クリックしてノードを配置し、名前を *work_item_viewer* に変更します。

File ノードは通常、ディスク上のファイルを指します。まずはこの方法でファイルを見つけ、その後、別の方法を使ってワークアイテムを TOP から直接取得します。



05 *work_item_viewer* ノードをダブルクリックしてその内側に入り、パラメータペインで **[Geometry File]** のファイル選択ダイアログボタンをクリックします。ファイル選択ダイアログで、**$HIP** をクリックしてから *geo* フォルダをダブルクリックします。

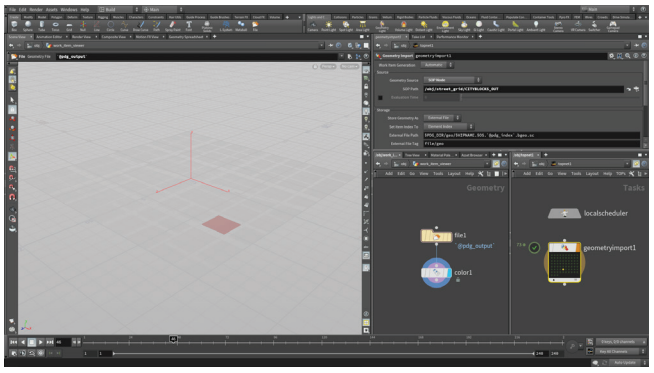
city_01_geometryimport ファイルシーケンスを選択します。すると、保存されていたジオメトリが、番号付きのシーケンスとして読み込まれます。ジオメトリシーケンスの後に **color** ノードを追加し、そのカラーを**赤色 (1, 0, 0)** に設定します。タイムラインでスクラブすると、さまざまなジオメトリが順番にロードされる様子を確認できます。ディスクに保存されていたジオメトリは、73 個です。



06 都市区画の表示を、フレーム番号にリンクする代わりに TOP ネットワークに直接リンクしてみましょう。**[Geometry File]** を次のエクスプレッションに変更します。

```
`@pdg_output`
```

これは、ディスクからファイルを順番にロードする代わりに、topnet でアクティブになっているワークアイテムをロードすることを意味します。

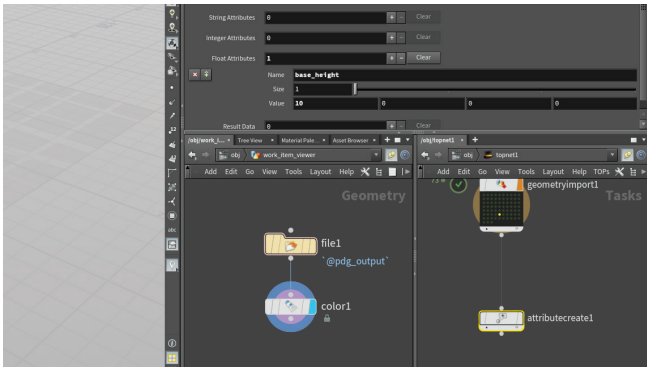


07 ここで topnet に戻り、*geometryimport* TOP ノードでワークアイテムのドットをクリックします。そのワークアイテムに関連するジオメトリがビューポートに表示されます。

作業内容を**保存**します。

パート 3:
アトリビュートを追加する

ビルを作成する際には、固定されたベースの高さとランダムな高さバリエーションを設定し、すべてのビルのサイズが同じにならないようにする必要があります。これらのアトリビュートは、都市マップのジオメトリレベルでセットアップすることも可能ですが、ここ（TOP）で割り当てることもできます。そうすれば、後で TOP レベルで変更を加える必要が生じても、容易に対処することができます。



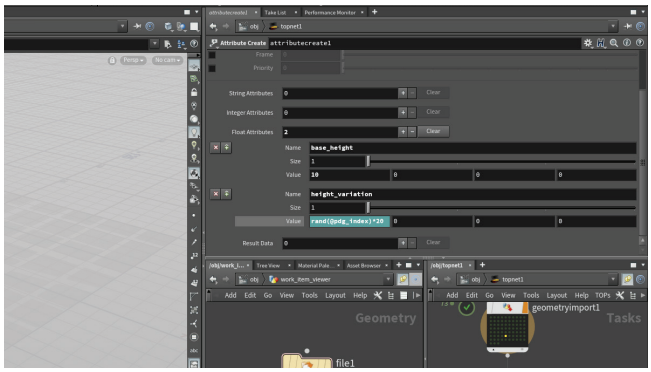
01

TOP ネット内で **Attribute Create** TOP を追加します。**[Work Item Generation]** を **[Dynamic]** に設定します。

[Float Attributes] の**プラス**記号をクリックして新しいアトリビュートを作成した後、以下の値を入力します。

- **[Name]**: *base_height*
- **[Value]**: `10`

これにより、すべてのビルが10以上の値を持つことが保証されます。最も低いビルの高さを変更する必要が生じた場合、後でこの数値を調整することができます。



02

プラス記号を再度クリックし、以下の値を入力します。

- **[Name]**: *height_variation*
- **[Value]**: `rand(@pdg_index) * 20`

この場合、ワークアイテムのアトリビュート **Index** をシードとして使用して、0 から 20 までの乱数が作成されます。ビル作成用ネットワークを作成する際には、この値をベースの高さの値に加算することで各ビルの高さを決定します。

03

attributecreate ノードを選択し、**Shift-G** キーを押してクリックします。ビルはまだ作成していないので、3D ビューには依然としてフットプリントが表示されます。

ワークアイテム上で**中マウスボタンをクリック**すれば、値 **10** の *base_height* と、**0 と 20** の間の数値となる *height_variation* が各ワークアイテムに含まれていることがわかります。アトリビュートは、このTOP ノードで生成されましたが、まだ使用されていません。

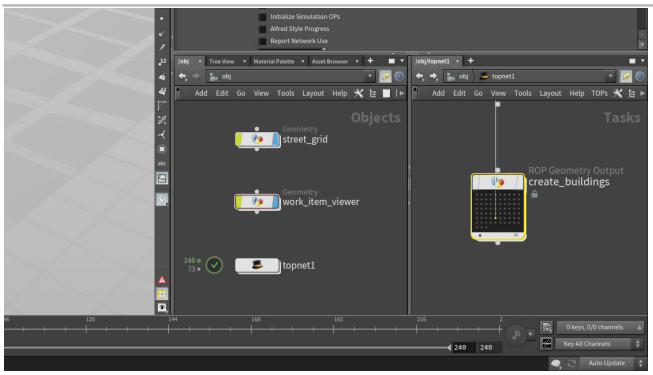
TOP でのアトリビュートの追加

都市グリッドのジオメトリネットワークでアトリビュートを追加することは以前から可能でしたが、TOPでそれらを追加すれば、TOPのコンテクストでそれらに変更を加えやすくなります。アトリビュートはパイプラインの全体に重要な情報を提供するための重要な手段なので、それらを適切にセットアップすることが重要となります。

```
pointcount 30
primitiveoffset 27
base_height 10.0
height_variation 15.2391648293
```

パート 4:
都市グリッドに対するビルを作成する

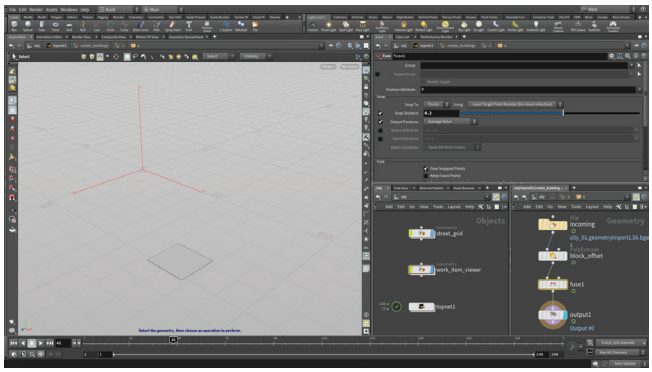
ここでは、ROP Geometry ノードを使って単純なビルを作成します。作業内容をビューポートで確認するには、ワークアイテムを生成し、そのいずれかを選択する必要があります。その後、そのワークアイテムを使ってジオメトリレベルでビルをデザインできます。



01 **ROP Geometry Output** TOP を追加します。その名前を *create_buildings* に変更します。**[Use External SOP]** オプションを**オフ**にします。

その上で**右マウスボタンをクリック**し、**[Generate node]** を選択してワークアイテムを作成します。これらは、まだ処理されていない灰色のワークアイテムです。これらは実際には、このノードのセットアップ後に実行されるタスクのプレースホルダです。

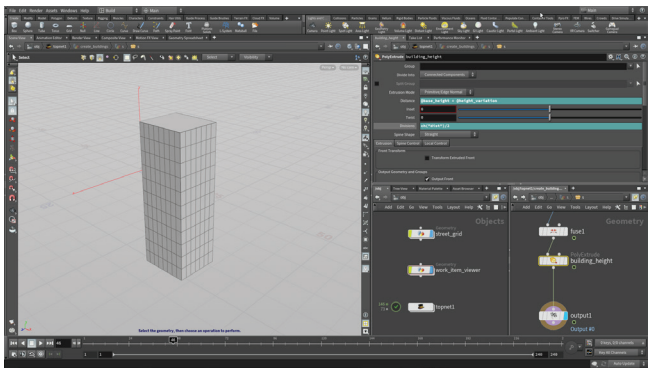
重要: いずれかのワークアイテムをクリックしてください。その場合、*work_item_viewer* ネットワークでエラーが生成されますが、それは、このノードでまだジオメトリが 1 つも生成されていないからです。



02 このノードを**ダブルクリック**してその内側に入り、ジオメトリレベルに移動します。

PolyExtrude ノードを作成し、これを *incoming* ノードと *output* ノードの間に配置します。その名前を *block_offset* に変更します。**[Inset]** を 1 に設定し、**[Extrusion]** の下にある **[Output Side]** オプションを**オフ**にします。こうすれば、ビルの歩道の作成が可能になります。

次に、この PolyExtrude の後に **Fuse** ノードを追加し、**[Snap Distance]**を**0.2**に設定することで、小さな線がすべて除去されるようにします。work_item_viewer ノードでエラーが発生しているのに、ビューポートに何かが表示されていることに注意してください。この問題は後で修正します。



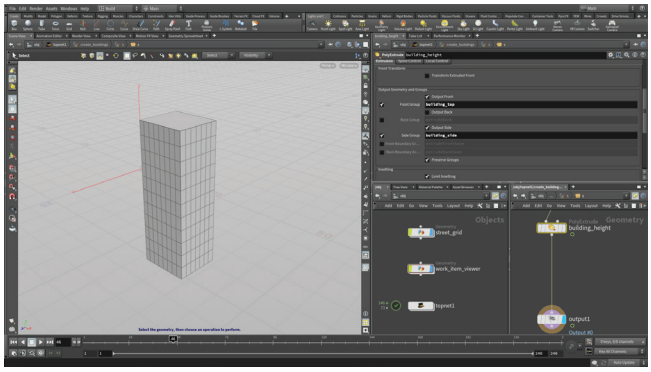
03 *fuse* の後に 2 番目の **PolyExtrude** ノードを作成します。その名前を *building_height* に変更し、**[Distance]** を次の値に設定します。

```
@base_height + @height_variation
```

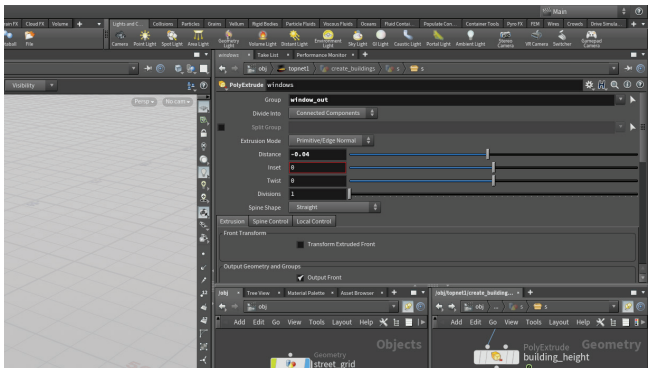
[Distance] パラメータが強調表示されていることを確認します。このパラメータで**右マウスボタンをクリック**し、**[Copy Parameter]** を選択します。**[Divisions]**パラメータで**右マウスボタンをクリック**し、**[Paste Relative References]** を選択します。結果のチャネル参照を編集し、次のように 2 で割ります。

```
ch("dist")/2
```

これで、床と窓のグリッドが表示されます。



04 [Extrusion] タブで、[Front Group] をオンにして名前を *building_top* にした後、[Side Group] をオンにして名前を *building_side* にします。これらは、ビルにディテールを追加する際に使用します。

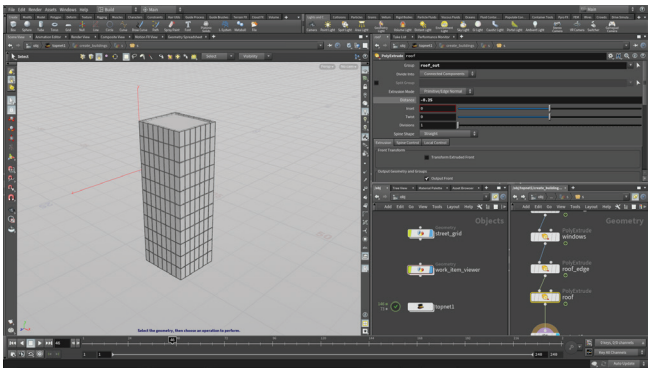


05

新しい PolyExtrude ノードをチェーンに追加し、その名前を *window_frames* に変更します。以下の設定を行います。

- **[Group]**: *building_side*
- **[Inset]**: 0.05
- **[Divide Into]**: **[Individual Elements]**
- **[Extrusion]** タブで、**[Front Group]** を**オン**にして名前を *window_out* にします。

新しい **PolyExtrude** ノードをもう 1 つ追加し、その名前を *windows* に変更します。その **[Group]** を *window_out* に設定した後、**[Distance]** を **-0.05** に設定します。

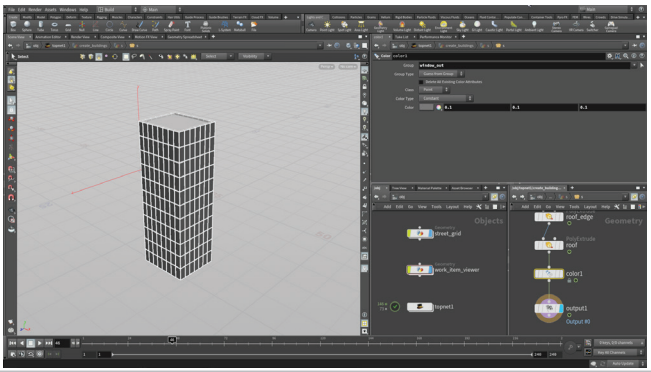


06

新しいPolyExtrudeノードをチェーンに追加し、その名前を *roof_edge* に変更します。以下の設定を行います。

- **[Group]**: *building_top*
- **[Inset]**: **0.3**
- **[Extrusion]** タブで、**[Front Group]** を**オン**にして名前を *roof_out* にします。

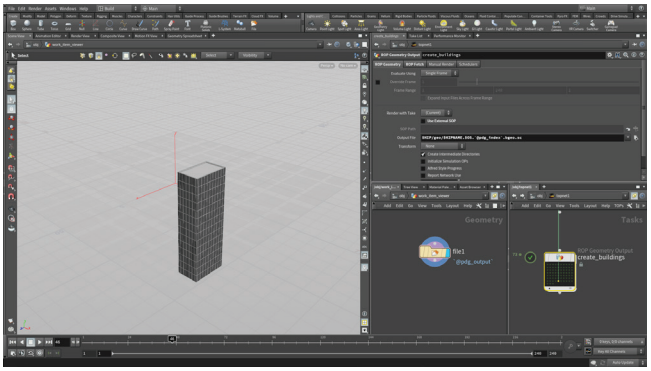
新しい PolyExtrude ノードをもう 1 つチェーンに追加し、その名前を *roof* に変更します。その **[Group]** を *roof_out* に設定した後、**[Distance]** を **-0.25** に設定します。



07

この後に color ノードを追加し、**[Group]** を *windows_out* に設定し、**[Color]** を**暗灰色 (0.1, 0.1, 0.1)** に設定します。以上のノードが *output* ノードに接続されていることを確認し、そのノードに**表示フラグ**を設定します。

これで、枠が明るい色、窓がより暗い色になります。これにより、ビューポートでのそれらの判読が容易になります。



08

上に 1 レベル戻り、ROP Geometry TOP で、**[Output File]** が次の値に設定されるように、`$F` を `` `@pdg_index` `` に変更します。

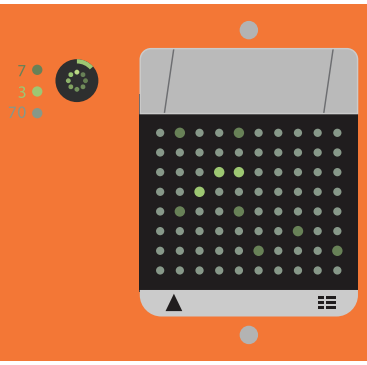
```
$HIP/geo/$HIPNAME.$OS.`@pdg_index`.bgeo.sc
```

この場合、フレーム番号の代わりに各ビルのインデックスファイルが使用されます。**[ROP Fetch]** タブをクリックし、**[Cache Mode]** を **[Write Files]** に設定します。シーンファイルを**保存**した後、*ropgeometry* ノードを選択し、**Shift-G** キーを押してクックします。

work_item_viewer の内側に戻り、もう赤色が表示されないように color ノードを削除します。ここで、create_buildings TOP ノードのワークアイテムをクリックすると、さまざまな高さのビルが表示されます。

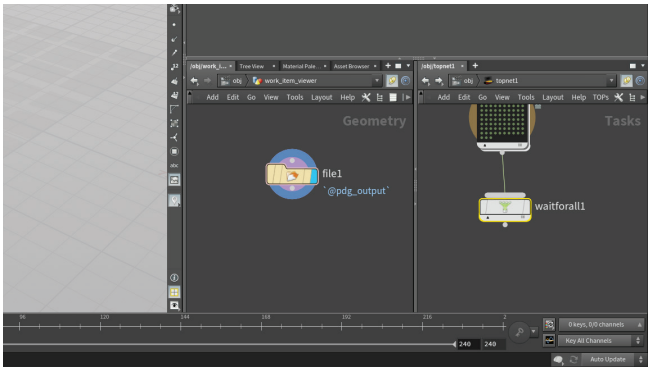
ワークアイテムの進行状況

これまで、ワークアイテムの処理はすぐに完了していましたが、このノードでは若干長い時間がかかります。終了したタスクを暗緑色で、進行中のタスクを黄色で、キュー内のワークアイテムを灰色で、それぞれ示す進行状況ホイールが表示されます。これを使えば、割り当てられたタスクがノード上で実行されていく様子を確認できます。

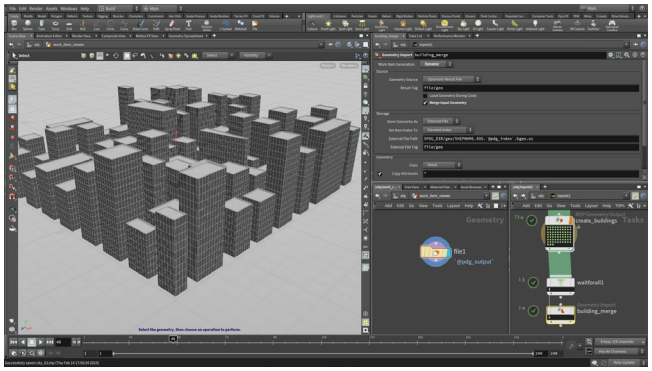


パート 5:
ビルを再結合する

ビルが完成したら、それらのビルを再度まとめて都市にする必要があります。それには、Wait for All と呼ばれるパーティションノードと、Geometry Import が必要になります。また、ワークアイテムを「ダーティ化」するような変更を行うので、再度クックする必要が生じます。

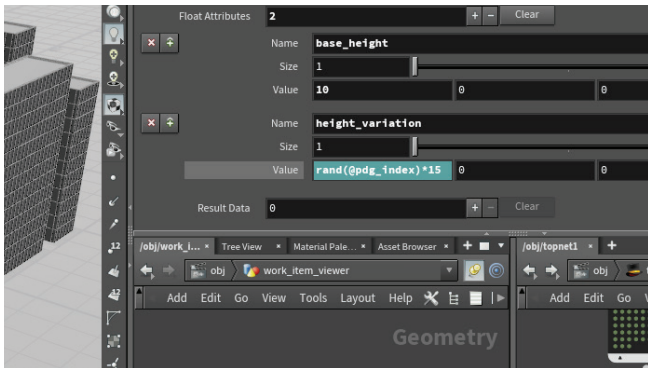


01 **Wait for All** TOP ノードを追加します。このノードは単純に、ROP Geometry ノードによって生成されたワークアイテムをすべて受け取り、それらを結合して単一のワークアイテムにします。さらにこのノードは、上側のすべてのワークアイテムのクックが完了するまで、下側の TOP が一切クックされないようにします。



02 **Geometry Import** TOP を追加します。その名前を *building_merge* にします。**[Work Item Generation]** を **[Dynamic]** に設定し、**[Merge Input Geometry]** オプションを**オン**にします。

シーンファイルを**保存**した後、*building_merge* ノードを選択し、**Shift-G** キーを押してクックします。ネットワークのクックが完了したら、最終的な *building_merge* ノードのワークアイテムをクリックします。ビューポートに都市の全体が表示されます。

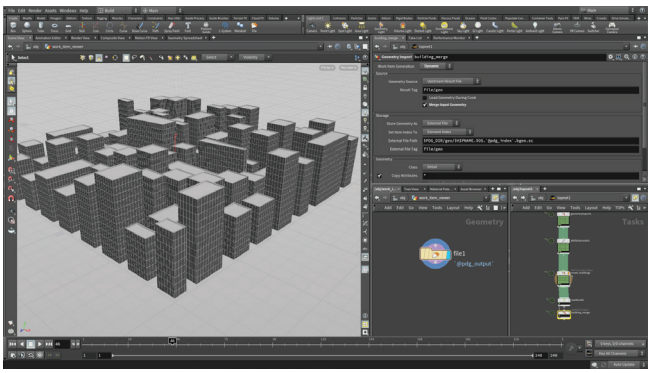


03 **Attribute Create TOP** に戻り、*height_variation* アトリビュートを次のように編集します。

- **[Value]**: `rand(@pdg_index) * 15`

これにより、ビルの高さが全体的にやや低くなります。

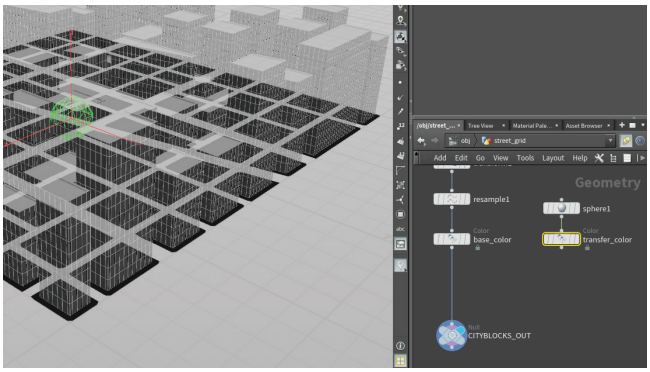
Attribute Create TOP で**右マウスボタンをクリック**し、**[Dirty Selected Node]** を選択します。すると、チェーン内の以降のワークアイテムがすべてクリアされます。



04 シーンファイルを**保存**します。次に、*building_merge* TOP ノードを選択し、**Shift-G** キーを押してクックします。ネットワークのクックが完了したら、最終的な *building_merge* ノードのワークアイテムをクリックして変更内容を確認します。

パート 6:
都市中心部を作成する

都市には明確な中心部が必要ですが、何らかのカラーを球から都市区画へトランスファーすることによって、これを実現できます。その場合、球をあちこちに移動させることで、都市中心部のさまざまな位置を試すことができます。またそれにより、変更発生時にダーティになったタスクを TOP ノードがどのように認識するかも理解できます。

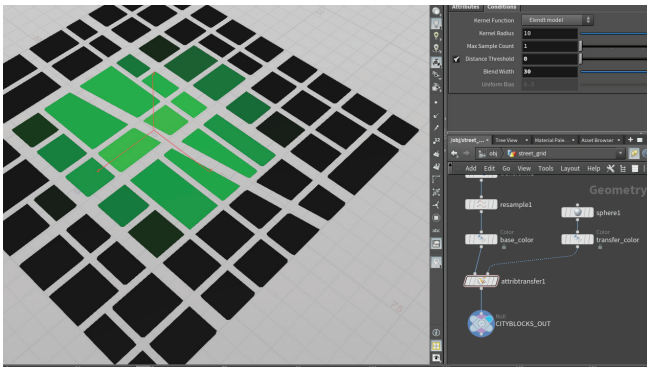


01 street_grid オブジェクトに移動し、その内側に入ります。**color** ノードを作成し、これをチェーン内の resample ノードの後に接続します。**[Class]** を **[Primitive]** に設定し、**[Color]** を **黒色 (0, 0, 0)** に設定します。このノードの名前を *base_color* に変更します。

street_grid ネットワーク内で球を作成し、端のほうにずらして配置します。以下のように設定します。

- **[Primitive Type]**: **[Polygon]**
- **[Uniform Scale]**: **5**

sphere の出力に Color ノードを追加し、**[Class]** を **[Primitive]** に設定し、**[Color]** を **緑色 (0, 1, 0)** に設定します。このノードの名前を *transfer_color* にします。

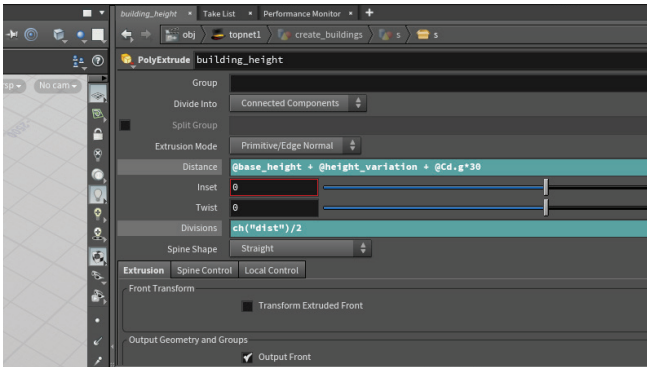


02 *base_color* ノードの後に **Attribute Transfer** ノードを追加します。続いて、その 2 番目の入力に *transfer_color* ノードを接続します。

[Points] チェックボックスのチェックを外し、**[Primitives]** フィールドを **Cd** に設定します。次に、**[Conditions]** タブで以下のように設定します。

- **[Distance Threshold]**: **0** に下げる
- **[Blend Width]**: 約 **30**

タイルが球の中心に近づくにつれ、徐々に緑色がタイルに浸透していくことがわかるはずです。



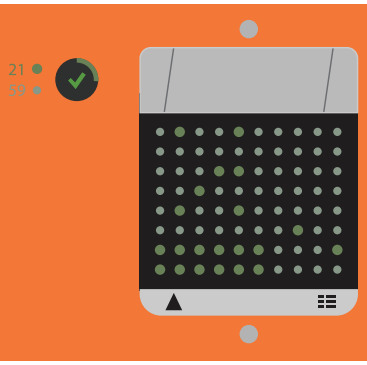
03 TOP ネットに戻って *attributecreate* ノードを選択し、*height_variation* アトリビュートを次のように編集します。

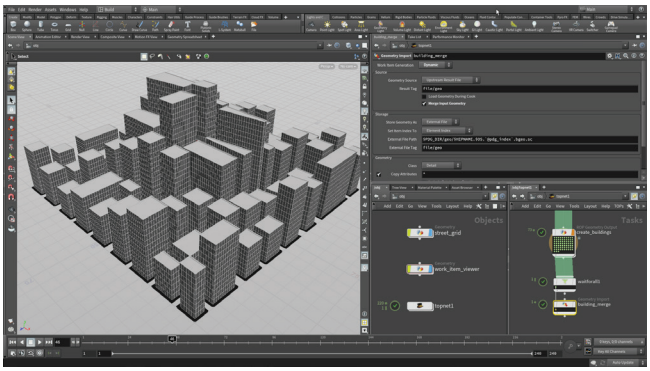
- **[Value]**: `rand(@pdg_index) * 15 + @Cd.g*30`

これにより、都市中心部のビルの高さが高くなります。最大で値30だけ高くなります。ビルの高さを変えたい場合はこの値を変更します。

ダーティ/クリーンなワークアイテム

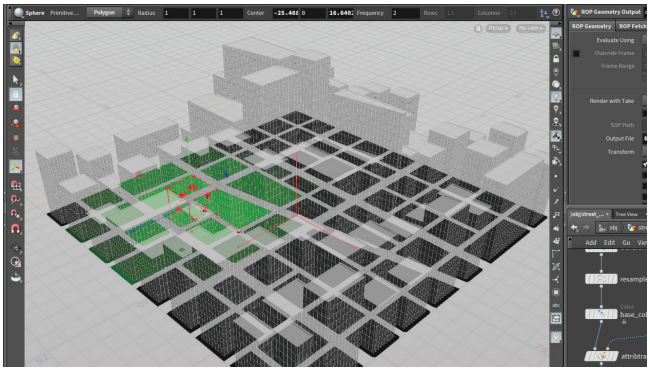
PDG テクノロジの主な目的の 1 つは、複雑なシステムで発生する依存関係です。都市の中心部に変更を加えると、ダーティになり再クックが必要となるノードと、そのままで問題のないノードに分かれます。TOP におけるこうした依存関係の処理方法が、TOP の強みの 1 つです。



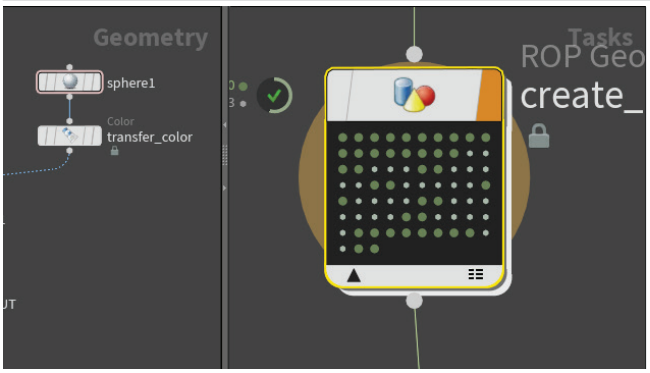


04 *geometryimport* ノードで**右マウスボタンをクリック**し、メニューから **[Dirty Selected Node]** を選択します。あるいはこのノードを選択し、**Shift-D** キーを押してダーティ化してもかまいません。

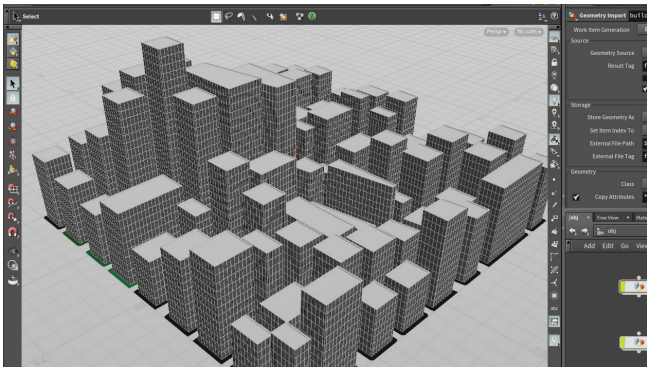
building_merge ノードを選択し、**Shift-G** キーを押してクックします。準備が整ったら、最終的な *building_merge* ノードのワークアイテムをクリックします。都市中心部の更新を含む都市が、ビューポートに表示されます。



05 *street_grid* オブジェクトに移動し、*CITYBLOCK_OUT* ノードが表示された状態で *sphere* ノードを選択し、**ハンドル**ツールを使ってこのノードをあちこちに移動させると、都市グリッドのさまざまな部分にその影響が及ぶことがわかります。



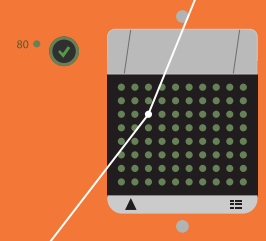
06 TOP ネットに戻り、*geometryimport* ノードを確認します。ネットワークビューの **[TOPS]** メニューから **[Generate Static Work items]** を選択します。ジオメトリレベルで行われた変更のため、一部のワークアイテムが自動的に「ダーティ化」されたことがわかります。残りはまだクリーンとみなされるので、再生成の必要はありません。



07 ここで、*building_merge* ノードで**右マウスボタンをクリック**して **[Cook Selected Node]** を選択した場合、ダーティなワークアイテムのみが更新されます。ワークアイテムをクリックすると、新しい都市中心部が表示されます。これが、ワークフロー開発時に TOP の依存関係グラフが効率的に動作する方法の 1 つです。

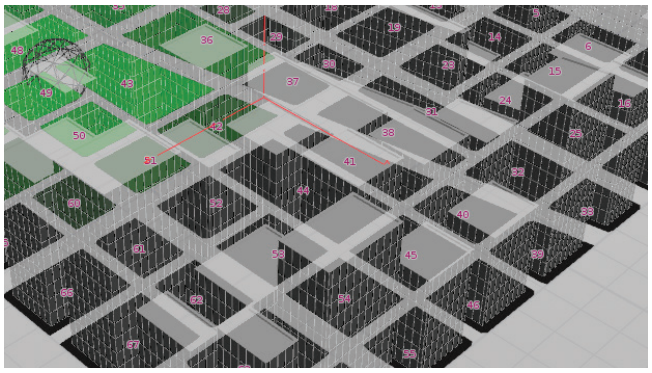
更新される依存関係

あるワークアイテムを選択すると、そのワークアイテムがシステム内の他のワークアイテムにどのように接続されているかを示す線が表示されます。これは依存関係のマッピングであり、これによってグラフの動作が効率化されます。なぜなら、**Shift-D** キーでノード上のすべてのワークアイテムを明示的にダーティ化しない限り、ダーティなワークアイテムのみが処理されるからです。



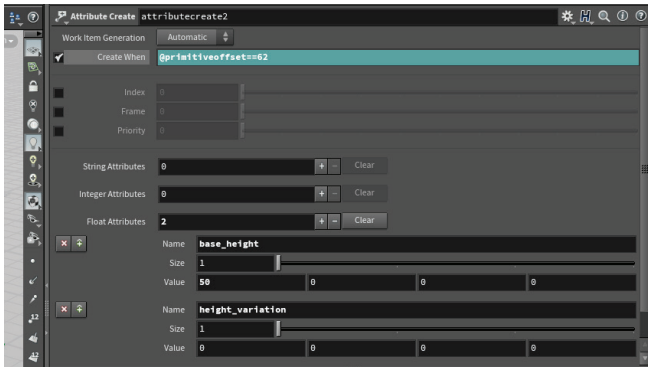
パート 7:
1 つのビルにフォーカスする

これまで、ビルの高さは height_variation のランダム性によって決定されていました。特定の高さに固定したいビルが 1 つ存在する場合、2 番目の attribute create ノードを使ってプリミティブの選択と特定の値の設定を行うことができます。この手法を使えば、ランダム性をオーバーライドしたい場合に、システムに関する芸術的な指示を行いやすくなります。



01 *street_grid* オブジェクトに戻り、表示オプションの [Display Primitive Numbers] をオンにします。区画番号が表示されます。これで、各ビルを識別し、そのいずれかを明示的に設定するかどうか判断できるようになりました。**区画 62** のビルの高さを高くしてみましょう。

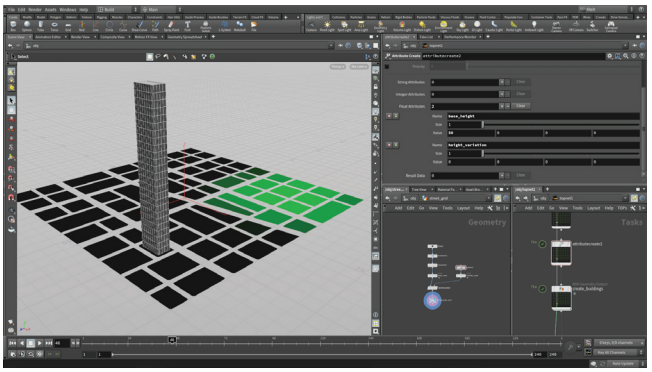
現在の設定のランダム性に依存する代わりに、この都市区画用の特定のパラメータを設定し、望みどおりの正確な高さが得られるようにします。これにより、ランダム性に基づいてすべてを決定するのではなく、TOP ネットワーク内で何らかの芸術的な制御を行えるようになります。



02 ここで、*attributecreate* TOP ノードで **Alt キーを押しながら**ドラッグすることで、そのコピーを作成します。まだ接続はしないでください。**[Create When]** の横にあるチェックボックスをオンにし、次のエクスプレッションを追加します。

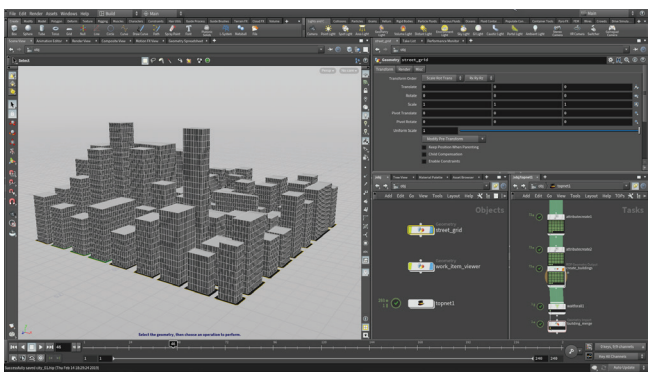
@primitiveoffset==62

height_variation の **[Value]** パラメータで**右マウスボタンをクリック**して **[Delete Channels]** を選択した後、その値を **0** に設定します。その後、*base_height_* の値を **50** に設定します。これで、**区画 62** のビルの高さが明示的にちょうど 50 単位になります。



03 このノードを、最初の *attributecreate* と *create_buildings* ノードの間に挿入します。作業内容を**保存**した後、*building_merge* ノードを再クックします。

create_buildings ノード上のインデックス 62 のワークアイテムをクリックすると、区画 62 のビルの高さが 50 単位で表示されます。

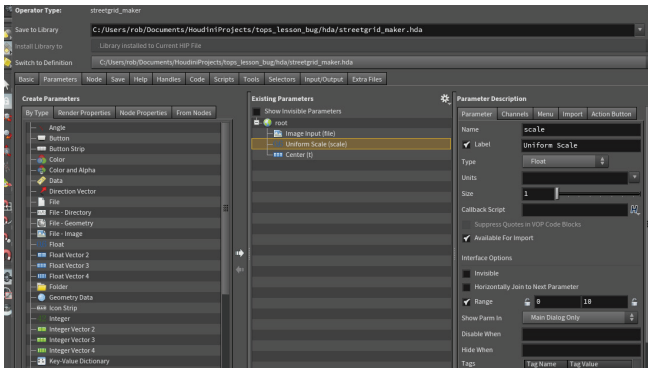


04 *building_merge* ノードのワークアイテムをクリックすると、新しいビルがそびえ立つ新しい都市景観が表示されます。これで、ユーザがネットワークをクックし直しても区画 62 のビルは更新されず、明示的に設定された状態に保たれるようになります。

そのビルを変更したい場合は、2 番目の attribute create ノードを使ってその高さを明示的に設定することができます。

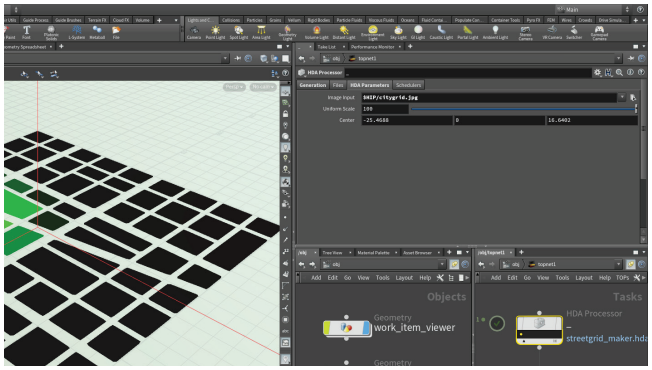
パート 8:
都市中心部の位置をウェッジする

street_grid オブジェクト内での球の位置をウェッジするには、そのネットワークを Houdini Digital Asset [HDA] に変換し、HDA Processor TOP ノード経由で実行する必要があります。これらのアセットをロードしてシステムに追加できれば、必要に応じて更新/適応可能な個別のツールから、複雑なシステムを作成することが可能となります。



01 *street_grid* ネットワーク内に移動して *CITYBLOCKS_OUT* 以外のすべてのノードを選択し、**[Assets]** メニューから **[New Digital Asset from Selection]** を選択します。アセットの名前を **streetgrid_maker**、ラベルを **Street Grid Maker** にした後、アセットを `$HIP/hda/` ディレクトリに保存します。

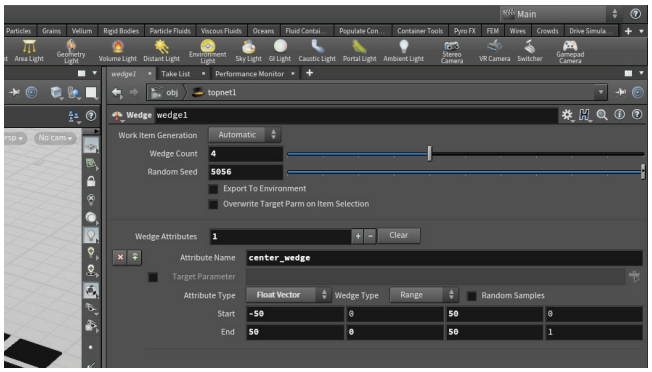
ポップアップウィンドウで **[Parameters]** タブをクリックします。ネットワークビューでこのアセットの内側に入り、*trace* ノードを選択します。**[Image Input]** パラメータをパラメータリストにドラッグします。*transform* ノードを選択し、**[Uniform Scale]** パラメータをドラッグします。*sphere* ノードを選択し、**[Center]** パラメータをドラッグします。**[Accept]** をクリックします。



02 TOP ネットワークに戻ります。**HDA Processor TOP** を作成し、**Geometry Import** ノードに接続します。*geometryimport* ノードで、**[Work Item Generation]** を **[Dynamic]** に、**[Geometry Source]** を **[Upstream Result File]** にそれぞれ変更します。

HDA Processor ノードを選択し、**[HDA File]** を *$HIP/hda/streetgrid_maker.hda* に設定します。**[HDA Parameters]** タブをクリックし、**[Image Input]** および **[Center]** パラメータが表示されていることを確認します。

この TOP の名前を *make_citygrid* に変更した後、**Shift-V** キーを押してノードの「ダーティ化とクック」を実行し、このノードがまだ以前と同じ方法で都市グリッドを生成していることを確認します。



03 **Wedge** TOP ノードを作成し、*make_citygrid* に接続します。以下の設定を行います。

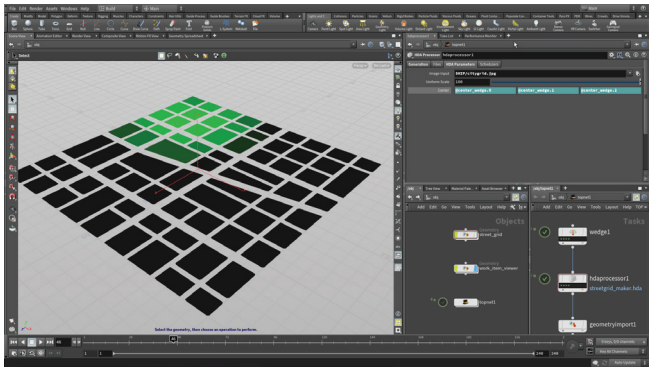
- **[Wedge Count]: 4**

[Wedge Attributes] の横にある**プラス記号**をクリックし、最初のアトリビュートの **[Attribute Name]** を *center_wedge* に設定した後、以下の設定を行います。

- **[Type]: [Float Vector]**
- **[Start] (範囲): -50, 0, -50, 0**
- **[End] (範囲): 50, 0, 50, 0**
- **[Random Samples]: オン**

ウェッジング

ウェッジングは写真撮影に由来する用語であり、さまざまな設定を使って撮影しておき、後で現像所で最良のものを選べるようにする、というアイデアのことです。ここでも考え方は同じですが、ランダム値を使ってシステムに影響を与え、比較可能な固有の結果を 4 つ取得する、という点が異なります。後で、ランダム値が記載されたすべてのオプションを含むモザイクをレンダーします。

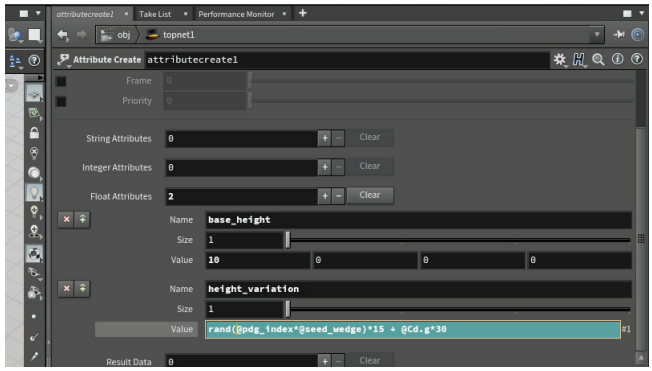


04

HDA Processor ノードを選択し、**[HDA Parameters]** タブの **[Center]** パラメータを次の値に設定します。

```
@center_wedge.0, @center_wedge.1, @center_wedge.2
```

make_citygrid HDA Processor ノードを選択し、**Shift-V** キーを押してノードの「ダーティ化とクック」を実行します。今度は 4 つの都市グリッドが作成されます。各ワークアイテムをクリックすると、それぞれ異なるグリッドが視覚化されます。緑色の位置（都市中心部の配置場所）はグリッドごとに異なります。



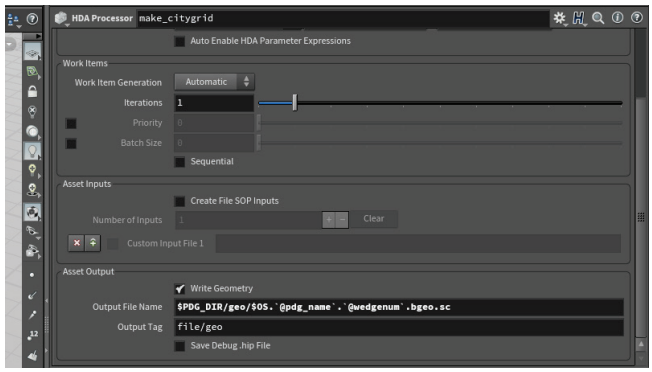
05

次に、高さのバリエーションにウェッジングを追加し、異なる4つの外観が得られるようにしましょう。*wedge* TOPを選択し、**[Wedge Attributes]** の横にある**プラス記号**をクリックし、2 番目のアトリビュートの **[Attribute Name]** を *seed_wedge* に設定し、**[Type]** は **[Float]** のままにし、**[Start]/[End]** を **0, 1000** に設定します。

[Random Samples] パラメータを**オン**にします。

最初の*attributecreate* TOPで、*height_variation* アトリビュートの**[Value]** を次のように変更します。

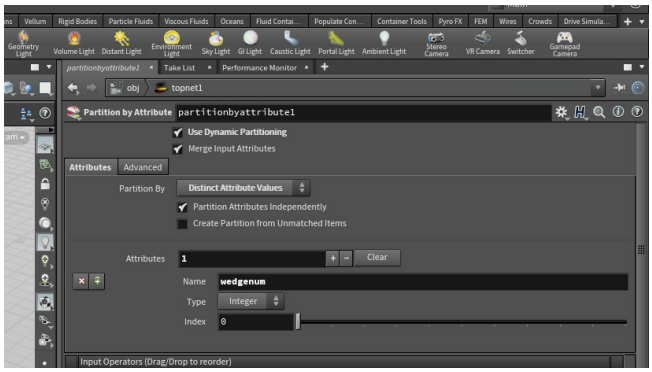
```
rand(@pdg_index*@seed_wedge) * 15 + @Cd.g*30
```



06

出力ファイルを作成するすべての TOP について、出力ファイル名にウェッジ番号が含まれるように更新する必要があります。そうすれば、ウェッジの異なるファイル同士を区別できます。そのために、以下の TOP ノードの出力ファイルパラメータで、`.bgeo` の直前に `.`@wedgenum`` を挿入してください。

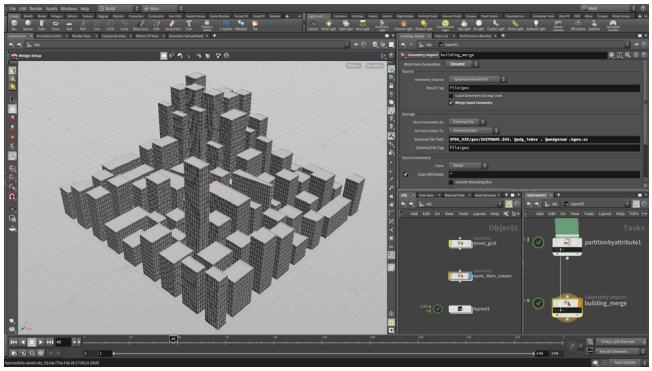
- *make_citygrid* HDA Processor
- *geometryimport* Geometry Import
- *create_buildings* ROP Geometry
- *building_merge* Geometry Import



07

Wait for All を **Partition by Attribute** TOP ノードに置き換えます。**[Partition By]** を **[Distinct Attribute Values]** に設定します。

[Attributes]の横にある**プラス記号**をクリックし、**[Name]**を*wedgenum* に設定します。**[Type]** は **[Integer]** に設定されたままにします。これにより、パーティションがウェッジごとに作成されます。



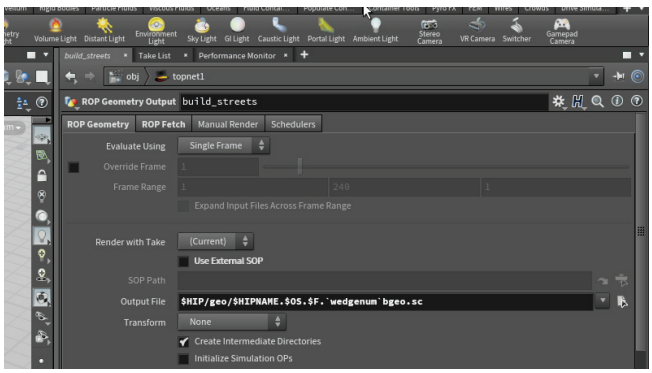
08

building_merge TOP ノードをクックします。すると、4 つの都市マップが作成されます。これらを視覚化するには、*building_merge* ノードのワークアイテムをクリックします。

作業内容を**保存**します。

パート 9:
街路のジオメトリを作成する

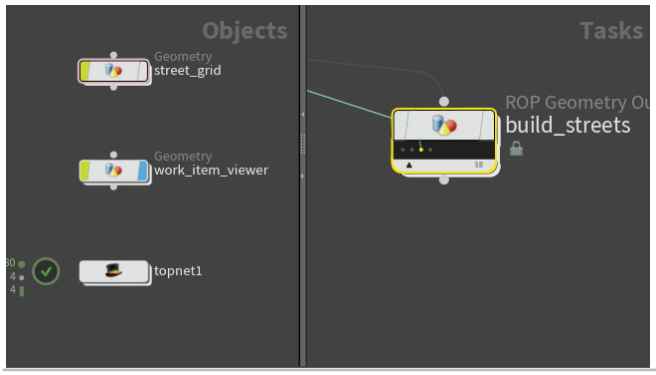
ここでは、ビルのコンテクストが得られるよう、最終的なレンダリングで使用すべき都市区画と街路を作成します。4 つのウェッジでは現時点ではすべて同じマップが使用されていますが、ここではこのジオメトリを TOP 内でセットアップします。これは、後でさまざまなマップを使用できるようにするためです。非常に堅牢なシステムを作成するには、柔軟性の高い方法で開発を進めることが常に推奨されます。



01 TOP ネットワーク内で **ROP Geometry** ノードを追加し、*make_citygrid* HDA Processor から分岐させます。この新しいノードの名前を *build_streets* に変更します。

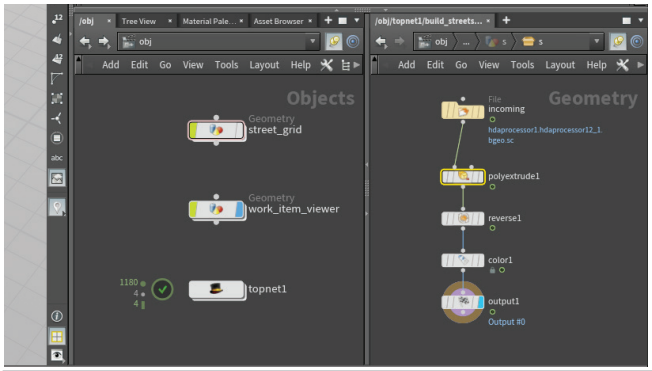
[Use External SOP] オプションを**オフ**にし、**[Output File]** で、エクスプレッションの `.bgeo` の直前に `.`@wedgenum`` を追加します。

[ROP Fetch] タブで **[Cache Mode]** を **[Write Files]** に設定します。



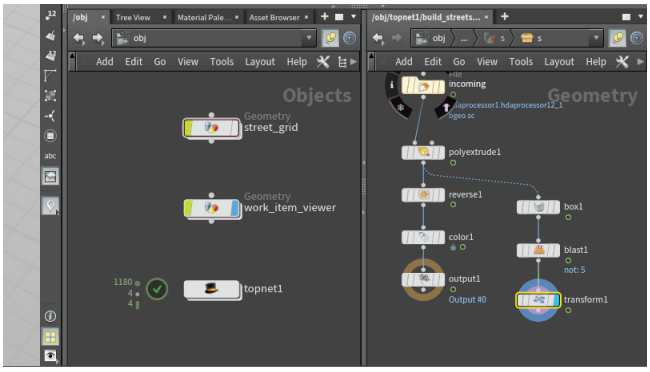
02 このノードで**右マウスボタンをクリック**して **[Generate Node]**を選択すると、4つのワークアイテムが得られます。これらのいずれかをクリックしてそのワークアイテムを強調表示させた後、ノードをダブルクリックしてその内側に入ります。

するとジオメトリレベルに移行するので、そこで街路を作成します。



03 *incoming* ノードと *output* ノードの間に **PolyExtrude** ノードを追加します。**[Distance]** を **-0.05** に設定します。**[Output Front]** チェックボックスを**オフ**にし、**[Output Back]** オプションを**オン**にします。

その後に **Reverse** ノードを追加して法線を修正した後、**Color** ノードを追加してすべての都市区画を白色にします。

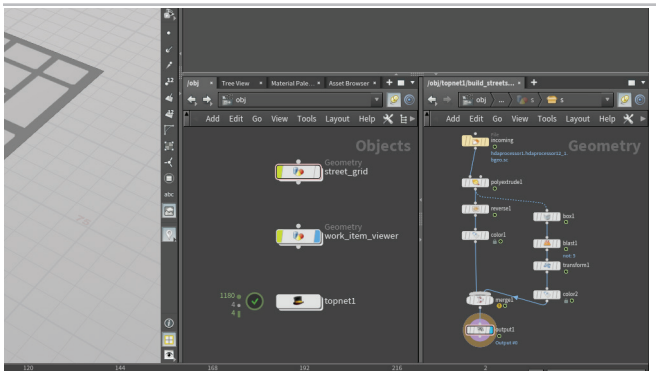


04 ネットワーク内の端のほうに **Box** ノードを追加します。*box* ノードに *incoming* を接続し、バウンディングボックスが都市グリッドのジオメトリに一致するようにします。

box の後に **Blast** ノードを追加し、**[Group]** を **5** に設定し、**[Delete Non Selected]** オプションを**オン**にします。

blast の後に **Transform** ノードを追加し、以下の設定を行います。

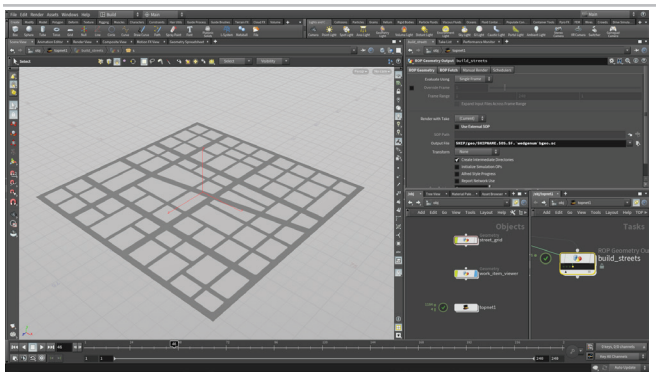
- **[Translate Y]**: **-0.05**
- **[Scale X]**: **1.05**
- **[Scale Z]**: **1.05**



05

ミディアムグレー (0.33, 0.33, 0.33) の **Color** ノードを追加します。

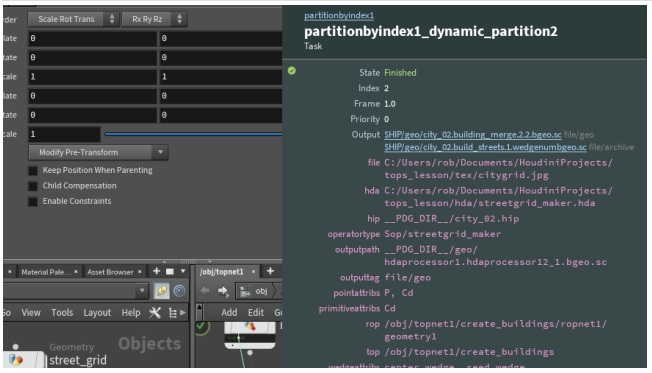
ここで Merge ノードを作成し、それを 2 つの *color* ノードに接続します。街路のジオメトリが画面に表示されるように、**表示フラグ**が出力に設定されていることを確認します。



06

シーンファイルを保存します。TOP ネットワークに戻ってノードをクックします。各ワークアイテムをクリックすると対応するグリッドが表示されます。

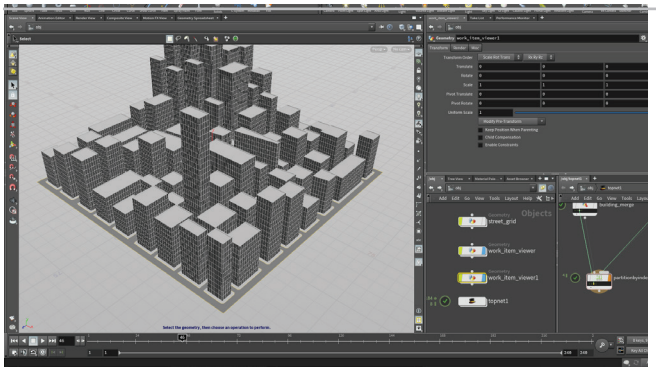
現時点では、これら 4 つはすべて同じです。これは、後でより多くの都市グリッドが導入された場合には異なってくる可能性があります。したがって、この例ではこのような方法でセットアップを行いました。



07

チェーンの末尾に **Partition by Index** を追加します。その最初の入力に *building_merge* ノードを接続し、2 番目の入力に **build_streets** を接続します。**[Use Dynamic Parenting]** オプションを**オン**にします。

新しいノードをクックします。処理の完了後、ワークアイテムをクリックすると、ビルのみが表示されていることがわかります。ワークアイテムを**中クリック**すると、Partition によって作成される出力ファイルは 2 つ存在することがわかります。新しい出力がビューポートに表示されるようにするには、新しい出力ファイルを表示してやる必要があります。



08

オブジェクトレベルに移動します。*work_item_viewer* で **Alt キーを押しながらドラッグ**して、2 つ目のビューアを作成します。ジオメトリレベルに入り、[Geometry File] を次の値に設定します。

```
`@pdg_output.1`
```

1 つ上のレベルのオブジェクトレベルに戻り、TOP ネットワークで *partitionbyindex* ノードのワークアイテムのいずれかを選択します。すると今度は、両方の出力がビューポートに表示されていることがわかります。このことは、次のステップで都市のイメージをレンダーする際に重要となります。

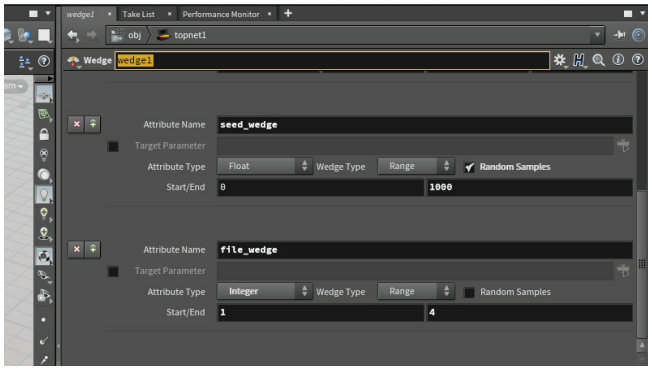
もう 1 つの PDG 出力

以前、各ノードからの出力結果を視覚化するのに役立つwork_item_viewerノードを作成する際には、pdg_outputを使用しました。しかし今ではこのパーティションが存在しているので、実際には2つの出力が存在します（ワークアイテム上で **Ctrl キーを押しながら中マウスボタン**をクリックすることで確認可能）。したがって、この 2 つ目の出力が正しく表示されるようにするには、独自のビューアが必要となります。

Output $HIP/geo/city_02.building_merge.2.2.bgeo.sc file/geo
$HIP/geo/city_02.build_streets.1.wedgenumbgeo.sc file/archive

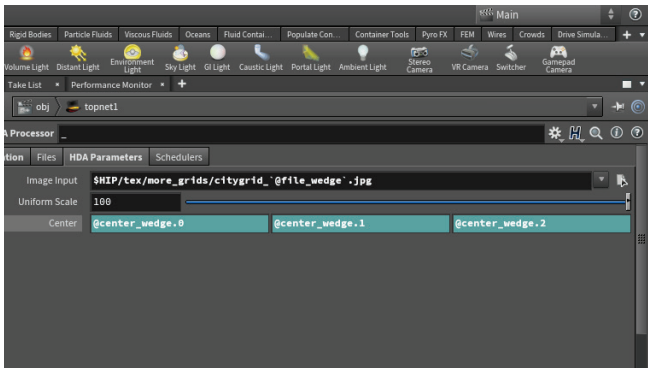
パート 10:
4 つの都市マップをウェッジする

現在のウェッジングに別の可変要素を追加してみましょう。異なる 4 つのマップにアクセスし、それらの各マップに対して 1 つのレンダリングを作成します。異なる白黒イメージのそれぞれがトレースされ、システムへの入力情報として使用されます。これは、パイプラインにコンテンツを追加するための別の方法を示しています。



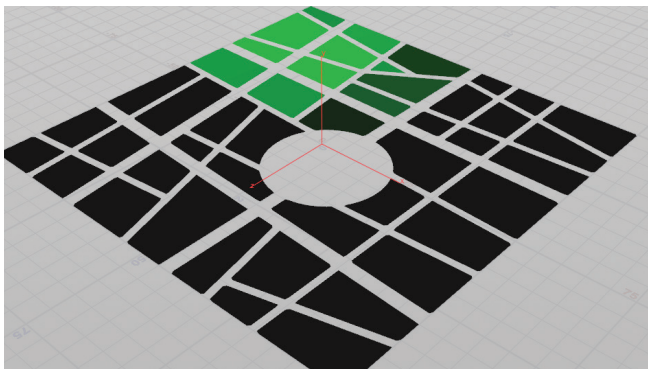
01 *wedge* TOP ノードに戻ります。[Wedge Attributes] の横にあるプラス記号をクリックして新しいウェッジパラメータを追加します。以下の設定を行います。

- **[Attribute Name]**: *file_wedge*
- **[Attribute Type]**: **[Integer]**
- **[Start]/[End]**: **1,4**

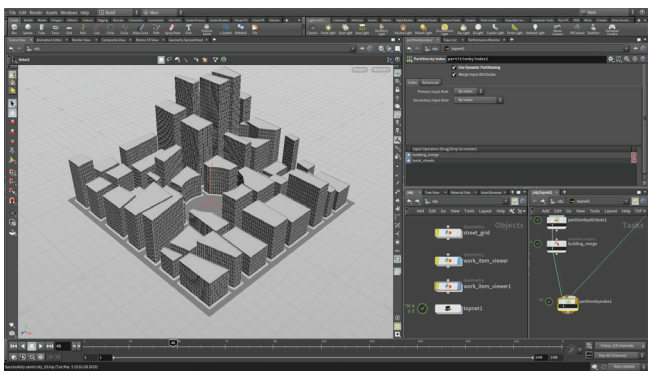


02 *streetgrid_maker* HDA Processor ノードで、**[Image Input]** の横にある **[File]** ボタンをクリックし、*more_grids* フォルダに移動します。そこからファイルを選択した後、パラメータペインで、エクスプレッション内の `$F` を `` `@file_wedge` `` に変更します。

このフォルダにはファイルが 4 つ存在していますが、ネットワークのクック時にそれらのファイルがすべて、このウェッジアトリビュートによって順番に処理されます。



03 *streetgrid_maker* HDA Processor ノードのダーティ化とクック（**Shift-V** キー）を実行します。異なる 4 つのマップが使用され、以前と同様に都市中心部があちこちに移動することを確認できます。

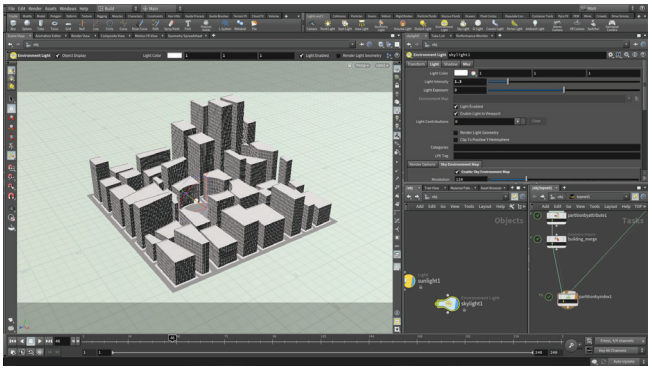


04 2 番目の attributecreate TOP ノードはバイパスします。これは静的なマップ向けに設計されたものであり、異なる 4 つのマップでは正しく動作しません。

最終的なパーティションノードのクック（**Shift-G** キー）を行い、4 つの都市計画が都市に変換される様子を確認します。

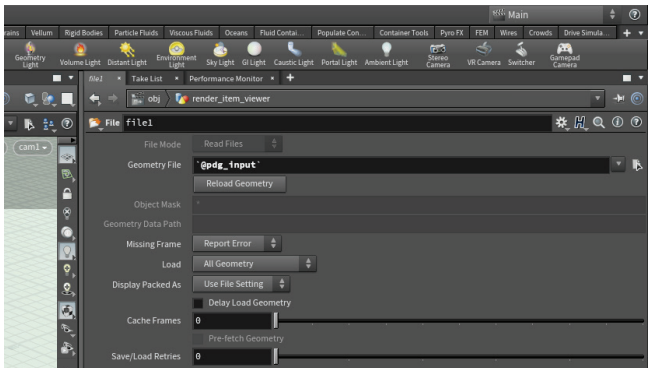
パート 11:
モザイクをレンダーする

ここでは、カメラやスカイライトを追加した後、都市をレンダーします。結果となるイメージは、ウェッジアトリビュートが記載された単一のモザイクにまとめられますが、これは、結果に基づいてクリエイティブな決定を下せるようにするためです。モザイクノードでは Image Magick アプリケーションが使用されるので、このアプリケーションがコンピュータにインストールされていないと、以下の手順を実行できません。



01 ある角度から都市を見下ろした状態になるまで、ビューを回転させます。カメラメニューから [New Camera] を選択します。4 つの都市グリッドをチェックし、どれもカメラに収まっていることを確認します。必要に応じて調整します。これはモザイクのレンダリング時に使用します。

Skylight を追加します。Environment Light の **[Intensity]** を **1.3** に設定します。



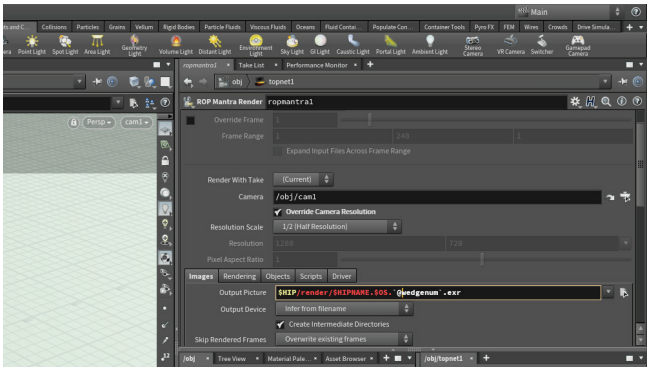
02 オブジェクトレベルで、*work_item_viewer* を **Alt キーを押しながらドラッグ**してコピーを作成します。新しいノードの名前を *render_item_viewer* に**変更**します。このノードの内側に入り、**[Geometry File]** パラメータを次の値に変更します。

```
`@pdg_input`
```

以上の手順を繰り返して *render_item_viewer1* を作成し、**[Geometry File]** パラメータを次の値に設定します。

```
`@pdg_input.1`
```

これらの新しい *render_viewer* ノードでは表示フラグを設定し、*work_item_viewer* ノードでは表示をオフにします。



03 *partitionbyindex* の後に **ROP Mantra Render** TOP を追加します。**[ROP Fetch]** タブで **[Cache Mode]** を **[Write Files]** に設定します。

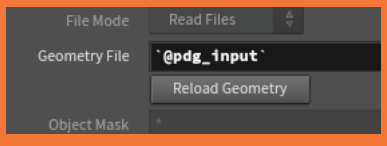
カメラのパラメータが先ほど作成したカメラに設定されていることを確認します。**[Override Camera Resolution]** オプションを**オン**にし、スケールを **[½]** に設定します。**[Output Picture]** パラメータを次の値に変更します。

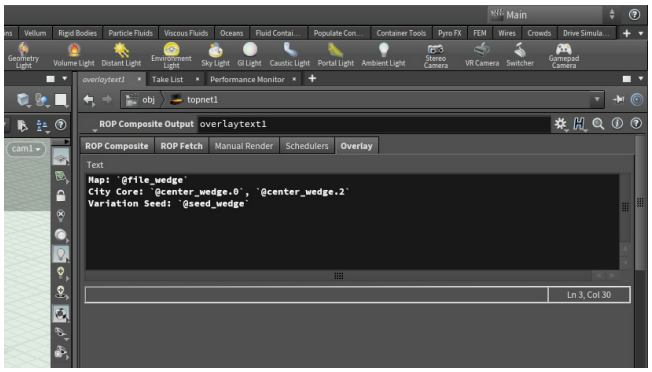
```
$HIP/render/city.$OS.`@wedgenum`.exr
```

注: Houdini Apprentice を使ってこのチュートリアルを実行している場合は、Apprentice のウォーターマークが表示されないように、`.exr` ファイルの代わりに `.pic` を使用するようにしてください。

PDG 入力

PDG 出力ではあるノードの結果を取得してそれを表示できるのに対し、PDG 入力は前のノードの結果を取得してそれを代わりに表示します。Mantra ノードはこの方法で、先行するノードの結果をジオメトリの入力として使用した後、そのジオメトリをレンダーすることができます。これら 2 つのオプションは似ていますが、その違いを理解しておくことが重要です。





04

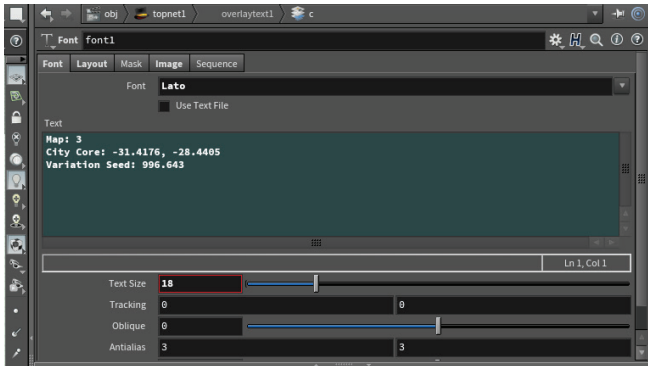
Overlay Text TOP を下に追加します。**[Cache Mode]** を **[Write Files]** に変更し、**[Output Picture]** を次の値に設定します。

```
$HIP/render/$HIPNAME.$OS.`@wedgenum`.exr
```

[ROP Fetch] タブで **[Cache Mode]** を **[Write Files]** に設定します。

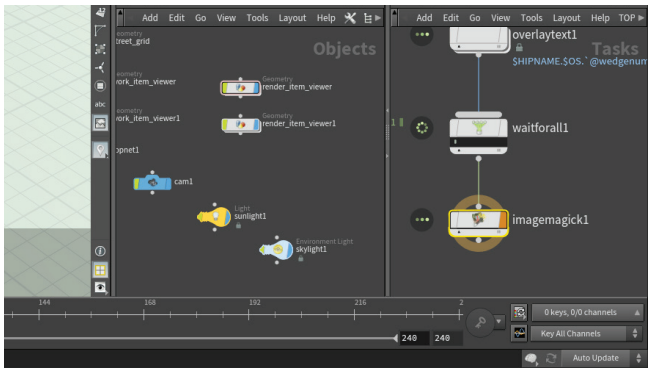
[Overlay] タブで次のように入力します。

```
Map: `@file_wedge`
City Core: `@center_wedge.0`, `@center_wedge.2`
Variation Seed: `@seed_wedge`
```



05

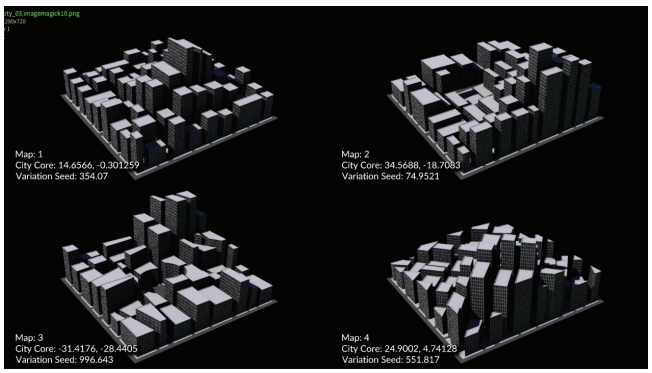
Overlaytext ノードを**ダブルクリック**してその内側に入ります。*Font* ノードを選択し、**[Text Size]** を **18** に変更します。こうすれば、作成されるテキストのサイズが小さくなり、都市の表示に利用できる領域が増えます。



06

上に戻って **Wait for All** パーティションノードを追加します。これはすべての要素を集約します。次に**ImageMagick** ノードを追加します。

このノードをクックして TOP ネットワークを処理します。すると、4 つのレンダリングが作成された後、テキストがオーバーレイされ、最後にそれらのレンダリングが 1 つのモザイクにまとめられます。



07

処理が完了したら、ワークアイテムで**右マウスボタンをクリック**し、**[View Task Output]** を選択します。すると、**Mplay** イメージビューアに最終的なイメージが表示されます。

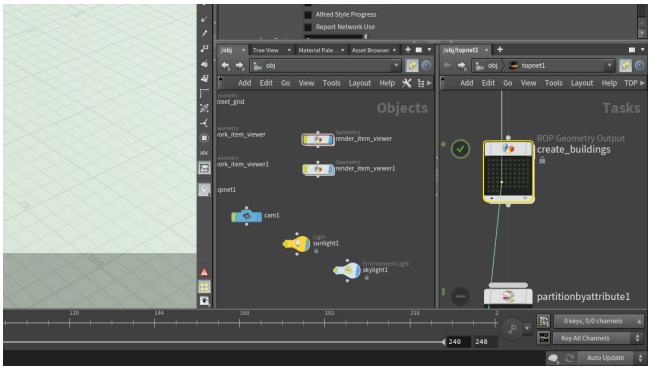
このコンタクトシートがあれば、さまざまなウェッジを確認したうえで、どのウェッジを採用するかを決定することができます。イメージ上に表示されたパラメータは、目的の都市を得るために確定する必要のある値です。

PilotPDG

TOP ネットワークの作成に特化した、PilotPDG という名前のスタンドアロンアプリケーションが存在しています。PilotPDG ではこのレッスンを作成できません。このアプリケーションの処理対象は TOP ノードだけなのに、ここではジオメトリネットワークの作成やライト/カメラのセットアップを行っているからです。ネットワークが完成したら、PilotPDG を使ってそのグラフを処理できます。PilotPDG では Houdini と同じ .hip フォーマットが使用されているからです。

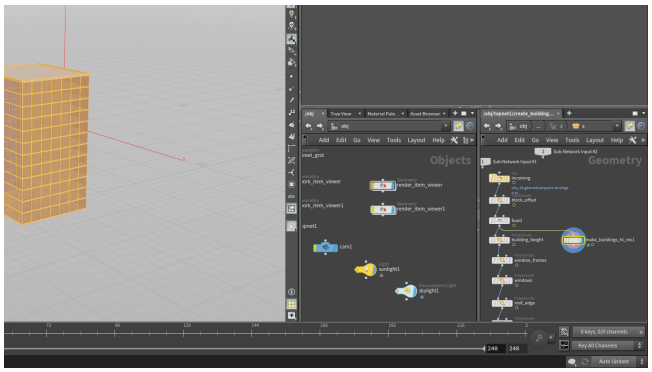
パート 12:
スケールアップしてより多くのコンテンツを作成する

この都市ビルサンプルはおそらく、さほど大きな問題もなく SOP で作成可能であると、最初に指摘しました。問題は、システムが複雑になるにつれ、ボトルネックが発生する点にあります。なぜなら、すべての処理が単一のネットワーク内で実行されるからです。TOPでは、コンピュートファームにワークアイテムを分散させることができるので、スケールアップ時の性能低下に悩まされることもありません。



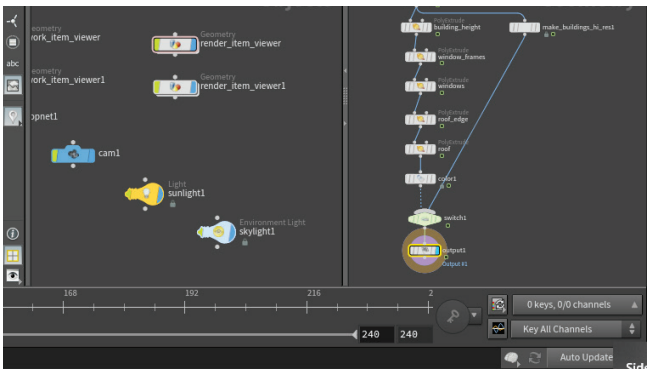
01 wedge TOP で [Wedge Count] を 1 に設定します。これで、スケールアップ中の反復回数が 1 回のみになります。

create_buildings ROP Geometry TOP ノードを選択し、**右マウスボタンをクリック**して **[Generate Items]** を選択します。次に、このノードの内側に入り、床、窓、柱を含む異なるタイプのビルを、デジタルアセットを使って作成します。



02 **ダブルクリック**して下のレベルに移動します。**[Assets]** メニューから **[Install Asset Library]** を選択します。*hda* ディレクトリに移動し、*make_buildings2.hda* を選択します。**[Accept]** をクリックした後、**[Install]** (**[Install and Create]** ではない) をクリックします。**タブキーを押して [Make Buildings Hi Res]** を選択します。グラフ内にノードが配置されます。

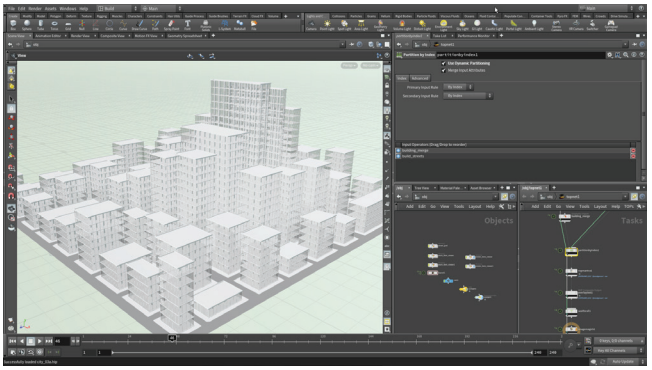
新しいアセットに *fuse* ノードを接続します。



03 *make_buildings_hi_res* デジタルアセットで以下の設定を行います。

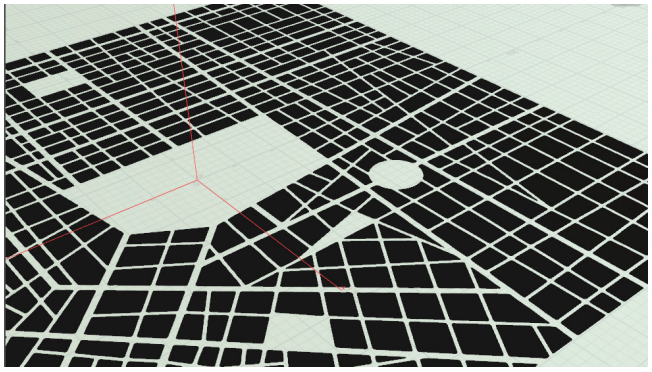
- **[Floor Height]**: **2**
- **[Base Building Height]**: `@base_height`
- **[Height Variation]**: `@height_variation`

switch ノードに接続し、switch ノードを 1 に設定します。



04 **[Partition by Index]** ノードを選択し、ネットワークを**クック**して新しいビルを表示させます。ビルに床版、柱、窓が含まれるようになりました。

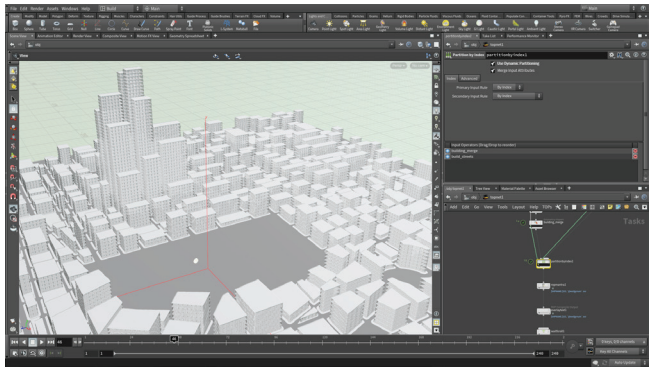
非常に多くのジオメトリが生成されています。ここで皆さんも、利用可能なワークアイテムの数をコンピュートファームを使って増やすのは良い考えかもしれない、と思い始めたのではないでしょうか。他のスケジューラノードの 1 つを使えばそれを実現できます。



05

また、別の都市グリッドイメージを使って都市の規模を大きくすることもできます。その場合、*streetgrid_maker* HDA Processor で、**[Uniform Scale]** を増やし、**[Image Input]** を新しい都市マップに変更する必要があります。

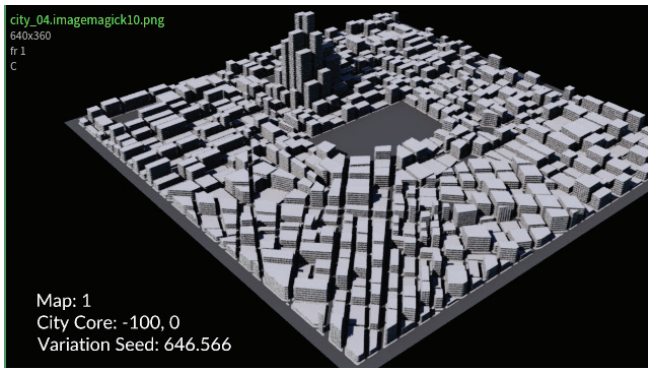
Wedge ノードで、新しいマップに合わせて *center_wedge* アトリビュートを設定することもできます。*streetgrid_maker* HDA Processor ノードの**ダーティ化とクック**を実行します。新しい都市マップが生成される様子を確認できます。



06

この処理が完了した時点で、元のマップに比べて極めて多くの都市区画が生成されているので、極めて多くのビルが生成されることになります。

この時点で、コンピュートファームの利用が重大な意味を持つはずです。なぜなら、単一のコンピュータでは TOP の並列処理機能を十分に活用できないからです。



07

最後のノードの**再クック**を実行して都市のイメージをレンダーします。我々が構築した単純なシステムが、必要なサイズに容易に拡張できる可能性を秘めていることに、皆さんも気づかれたはずです。

シーンを保存します。

まとめ

このレッスンでは、TOP ノードを使って都市マップを取得し、都市区画ごとにビルを作成した後、このシステムを拡張してより複雑なビルや大規模な都市マップを処理できるようにしました。このプロジェクトでは、TOP ベースのワークフローで使用される典型的なノードを紹介し、それらのノードを使ってワークアイテムのタスクを作成/処理する方法について説明しました。

TOP を使って作成可能なものは数多く存在しており、これはほんの始まりにすぎません。このネットワークタイプを使えば、Houdiniの典型的なワークフローを自動化/処理できるだけでなく、Houdini がまったく関与しないワークフローを処理することもできます。したがって TOP は、効率性を求める小規模なスタジオから、膨大なデータを管理する大規模なスタジオまで、パイプライン全体にわたって依存関係を管理するための優れたツールとなるはずです。