09. artisocレシピブック
3Dモデルを作成しよう

本ドキュメントについてのご質問、『複雑系勉強会』のお問合せは、下記までご連絡ください。

（株）構造計画研究所
社会デザイン・マーケティング部
artisocマーケティング担当　玉田
Tel: 052-222-8461
E-mail: tamada@kke.co.jp

ボイドモデルを3D化しよう

- ボイドモデルは鳥の群れのシミュレーションです。
- artisocをインストールすると、ボイドモデルがサンプルとして付いてきます。
- サンプルで群れの行動は確認できますが、二次元平面上を動くため、鳥っぽくありません。
- そこで、サンプルモデルを3Dモデル化する手順を説明します。

・ボイドモデルの拡張

① ボイドモデルとは？
② 空間を三次元にする
③ 3次元空間を出力する
④ いろいろな視点から三次元空間を見る
⑤ 建物に高さを与える
⑥ 鳥に高さを与える
⑦ 鳥の視点で飛んでみる
⑧ カメラ位置を指定する

① ボイドモデルとは？

■簡単なアルゴリズムで鳥の群れをシミュレーションしたい

- ボイド(Boids)は、アメリカのアニメーション・プログラマ、クレイグ・レイノルズが考案・作製した人工生命シミュレーションプログラムである。名称は「鳥もどき(bird-oid)」から取られている。

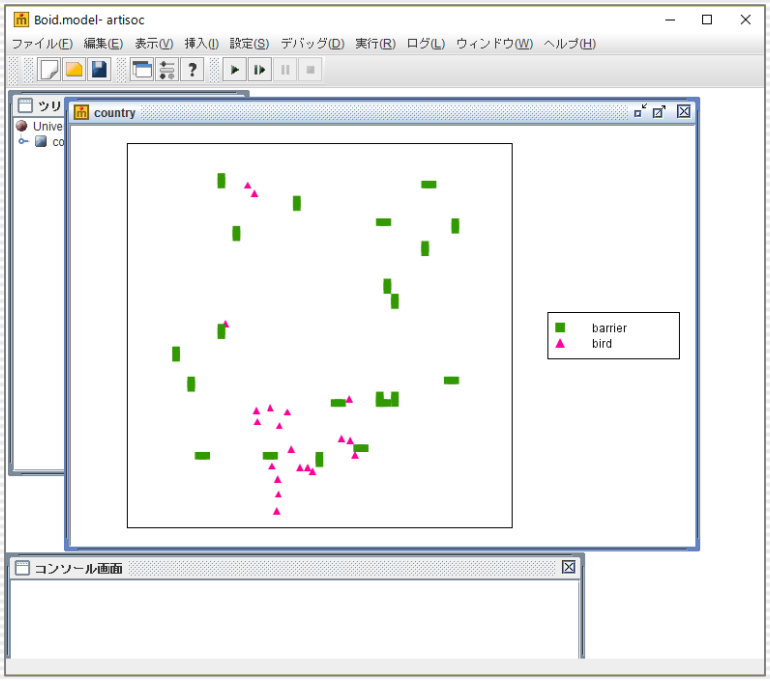
出典： https://ja.wikipedia.org/wiki/ボイド_(人工生命)

■ボイドモデルの3つの行動ルール

1. **分離（Separation）**
   鳥オブジェクトが他の鳥オブジェクトとぶつからないように距離をとる。

2. **整列（Alignment）**
   鳥オブジェクトが他の鳥オブジェクトと概ね同じ方向に飛ぶように速度と方向を合わせる。

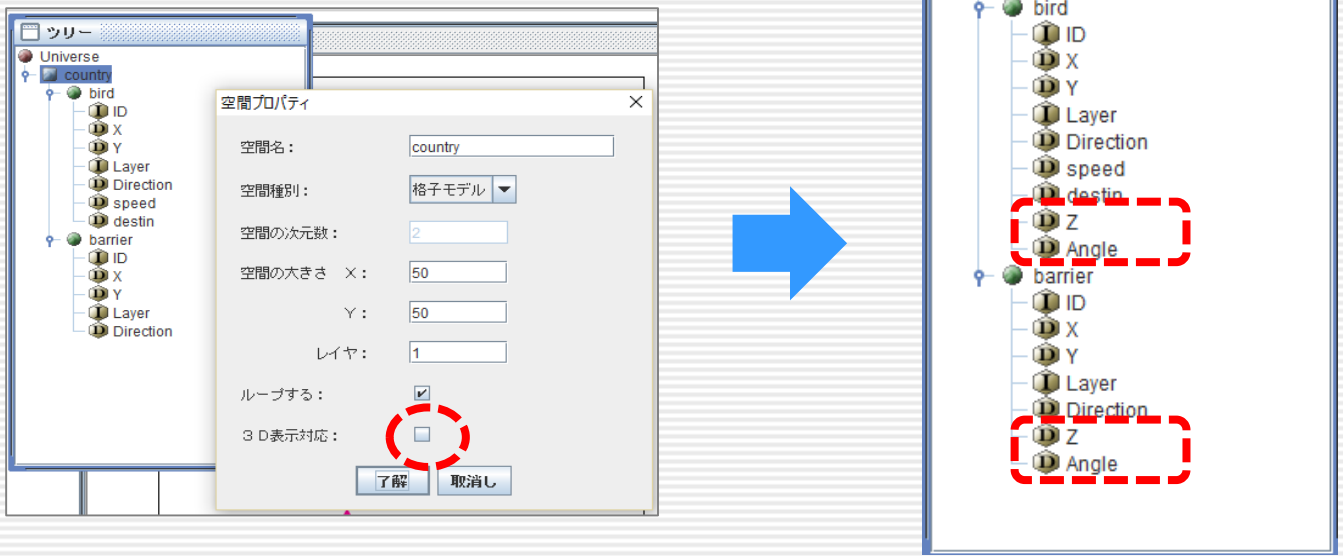
3. **結合（Cohesion）**
   鳥オブジェクトが他の鳥オブジェクトが集まっている群れの中心方向へ向かうように方向を変える。



ボイドモデル （▲が鳥、■が建物）
09-1.model ※artisoc同梱のBoid.modelと同じ

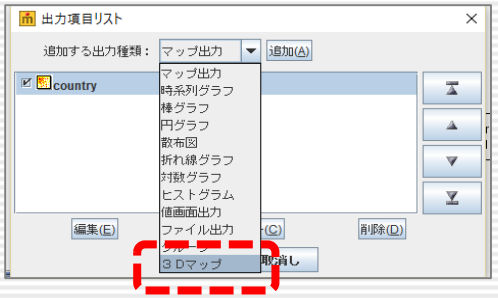
② 空間を三次元にする

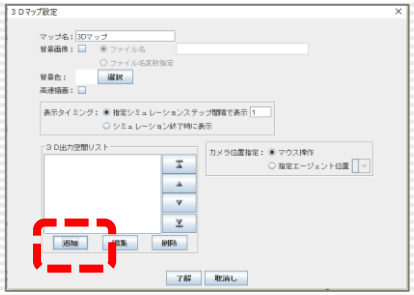
- 空間（country）のプロパティを開き、「3D表示対応」にチェックを入れます。
  - ツリーにて、「bird」「barrier」の変数として、「Z」「Angle」が追加されます。
  - 「Z」は高さ方向の座標です。「X」「Y」は二次元座標のときと同じく横、縦方向の座標です。
  - 「Angle」はエージェントの上下方向の傾きです。
    - カメラの向きを指定するときに利用し、詳しくは後述します。

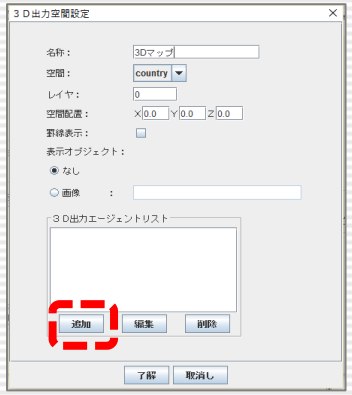


③ 三次元空間を出力する

- 設定メニューの「出力設定」にて、「3Dマップ」を追加します。
  - 3Dマップ設定ダイアログが表示されるので、「3D出力空間リスト」の「追加」をクリックします。
    - マップ名： 3Dマップ
  - 3D出力空間設定ダイアログが表示されるので、「3D出力エージェントリスト」の「追加」をクリックします。
    - 名称： 3Dマップ
    - 罫線表示： ON
  - 3D出力エージェント設定ダイアログが表示されるので、「bird」「barrier」を追加します。
    - 名称： bird
    - エージェント： bird
    - 表示オブジェクト： 三角柱
    - 表示色： 赤
    - 名称： barrier
    - エージェント： barrier
    - 表示オブジェクト： 四角柱
    - 表示色： 緑







④ いろいろな視点から三次元空間を見る

■シミュレーションを実行します。09-2.model

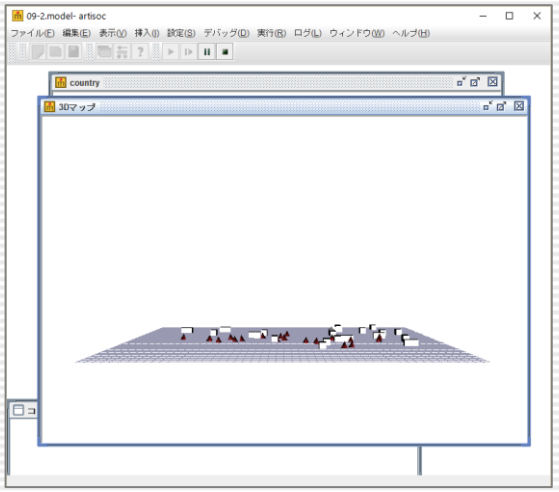
● 初期状態では、3Dマップはやや遠くに表示されます。

＜マウス操作＞

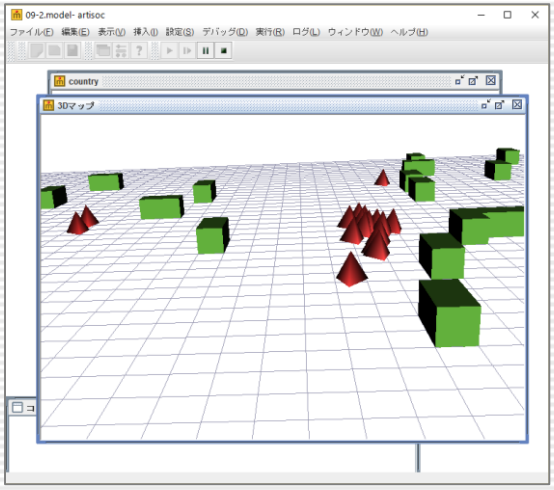
- 左ボタンを押しながら、前後左右に移動
- 右ボタンを押しながら、上下に移動
- 中ボタンを押しながら、アングルを上下に移動

＜キーボード操作＞

- ↑キー：　上方向を見る（アングル+）
- ↓キー：　下方向を見る（アングル−）
- ←キー：　左方向を見る（Direction−）
- →キー：　右方向を見る（Direction+）
- Sキー：　X軸方向に−
- Dキー：　X軸方向に+
- Eキー：　Z軸方向に+
- Xキー：　Z軸方向に−
- Wキー：　Y軸方向に+
- Qキー：　Y軸方向に−
- SPACEキー：Y軸方向に+[高速]
- RETURNキー：初期位置に戻る



初期位置はやや遠い視点



鳥の動きを近づいて見ることができる

⑤-1 建物に高さを与える

- シミュレーションを実行すると、三次元で表示できるものの、高さを与えてないので（Z=0）、鳥が地面を這うような動きになります。
- まずは、建物に高さを与えます。
  - 建物の高さ：　1～10
- ツリーの「Universe」を右クリックして、「ルールエディタ」を選択します。

```
Univ_Init{
    Dim Total as Agtset
    Dim One as Agt
    Dim Two as Agt
    Dim Three as Agt
    Dim i as Integer
    Dim j as Double
    Dim k As Integer
    Dim baseX As Integer
    Dim baseY As Integer
    Dim baseZ As Integer
```

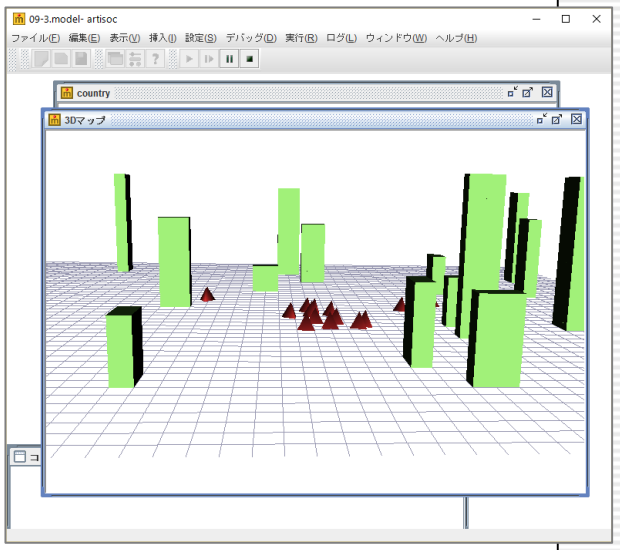
・・・建物の位置と高さを格納するための変数を定義する

⑤-2 建物に高さを与える

```
For i = 0 to 20
        baseX = Int(rnd()*40) + 6
        baseY = Int(rnd()*40) + 6
        baseZ = CInt(Rnd() * 10) + 1
        j = Int(rnd()*2)
        For k=0 To baseZ - 1
                One = createagt(Universe.country.barrier)
                One.X = baseX
                One.Y = baseY
                One.Z = k
                Two = createagt(Universe.country.barrier)
                Two.X = One.X + j
                Two.Y = One.Y + 1 - j
                Two.Z = k
                Three = createagt(Universe.country.barrier)
                Three.X = One.X + j/2
                Three.Y = One.Y + (1 - j)/2
                Three.Z = k
        Next k
Next i
}
```

・・・建物の高さを1～10でランダムに決める

・・・建物の高さまで
barrierを積み上げる



建物の高さが確認できる
09-3.model

⑥-1 鳥に高さを与える

- 次に、鳥を空に飛ばします。
  - 鳥の高さ：　1～5
- ツリーの「bird」を右クリックして、「ルールエディタ」を選択します。

```
Agt_Init{
    My.X = rnd()*50
    My.Y = rnd()*50
    My.Z = CInt(Rnd() * 5) + 1
    My.destin = rnd()*360
    My.speed = 0.2 + rnd()*0.3
}
```

・・・鳥の高さを1～5でランダムに決める

⑥-2 鳥に高さを与える

```
Agt_Step {

    ※省略

    For Each obj in Temp
        T = GetDirection(My.X, My.Y, obj.X, obj.Y, Universe.country)
        T = (T - My.Direction + 360) mod 360
        If 30 > T or 330 < T then
            If obj.Z - 1 <= My.Z Or My.Z <= obj.Z + 1 Then
                AddAgt(Objset, obj)
            End If
        End if
    Next obj

    If 0 < CountAgtset(Objset) then
        obj = GetAgt(Objset, Int(rnd()*CountAgtset(Objset)))
        If My.speed < obj.speed then
            My.destin = obj.destin
            My.speed = obj.speed
            My.Z = obj.Z
        End if
```

・・・自分の高さ±1の鳥のみ判定対象とする

・・・周りの鳥の高さに合わせる

⑥-3 鳥に高さを与える

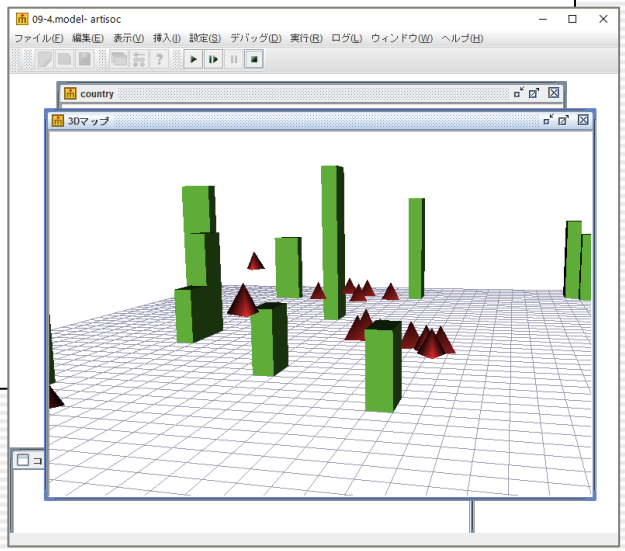
```
    Elseif 0 == CountAgtset(Temp) then
            My.destin = My.destin + rnd()*30 - 15
            My.speed = 0.2 + rnd()*0.3
            change_Z()                          ・・・周りに鳥がいないときは高さを変える
    End if

    ※省略
}
```

⑥-4 鳥に高さを与える

```
Sub change_Z()
{
    If Rnd() < 0.1 Then
        If Rnd() < 0.5 Then
            My.Z = My.Z - 1
        Else
            My.Z = My.Z + 1
        End If
    End If

    If My.Z < 1 Then
        My.Z = 1
    ElseIf My.Z > 5 Then
        My.Z = 5
    End If
}
```

・・・確率0.1で、上下方向に移動する

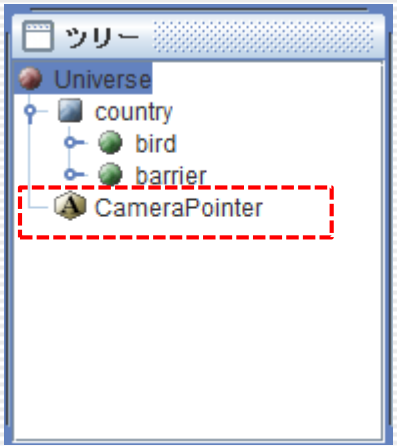
・・・鳥の高さ(1～5)をチェックする



建物の高さが確認できる
09-4.model

⑦ 鳥の視点で飛んでみる

- 鳥が空を飛ぶようになりましたが、手動で視点を移動させるのは面倒です。
- 鳥の視点でカメラを自動で切り替えます。
  - ツリーの「Universe」を右クリックして、「変数の追加」を選択します。
    - 変数名： CameraPointer：エージェント型
  - ツリーの「Universe」を右クリックして、「ルールエディタ」を選択します。

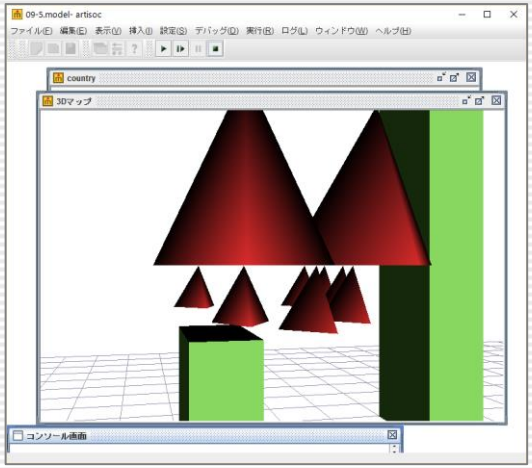



```
Univ_Init{

    ※省略

    Universe.CameraPointer = Universe.country.bird(0)
}
```

・・・鳥（ID=0）の視点をカメラ位置とする

  - 設定メニューの「出力設定」から「3Dマップ」の「編集」を選択します。
    - カメラ位置指定： 指定エージェント位置： CameraPointer

※カメラ位置はUniverse直下のエージェント型変数が選択できます。



  - シミュレーションを実行します。09-5.model
    - 鳥の視点で飛ぶことができます。

⑧-1 カメラ位置を指定する

- 鳥は視点だとせわしなく画面が切り替わってしまいます。
- カメラ位置をいくつか指定して、切り替えられるようにします。
  - ツリーの「country」を右クリックして、「エージェント型の追加」を選択します。
    - エージェント型名： Camera
    - 生成エージェント数： 1
  - ツリーの「Universe」を右クリックして、「変数の追加」を選択します。
    - 変数名： CameraPosition
    - 変数の型： 文字列型
  - ツリーの「Universe」を右クリックして、「ルールエディタ」を選択します。


```
Univ_Init{

    ※省略

    Universe.CameraPointer = Universe.country.Camera(0)
}
```

・・・カメラ（ID=0）をカメラ位置とする

  - 設定メニューの「コントロールパネル設定」を選択し「追加」をクリックします。
    - コントロール名： カメラ位置
    - 設定対象： CameraPosition
    - インタフェース： ドロップダウンリスト:リスト項目： 真上,ななめ,横

⑧-2 カメラ位置を指定する

- ツリーの「Camera」を右クリックして、「ルールエディタ」を選択します。

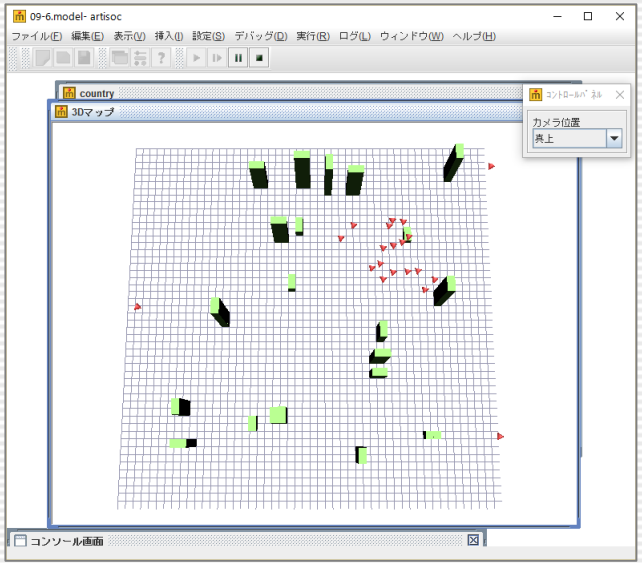
```
Agt_Step{
    If StrComp(Universe.CameraPosition, "真上") == 0 Then
        update_camera(25,10,80,90,80)
    ElseIf StrComp(Universe.CameraPosition, "ななめ") == 0 Then
        update_camera(25,-20,20,90,30)
    ElseIf StrComp(Universe.CameraPosition, "横") == 0 Then
        update_camera(25,-30,1,90,0)
    End If
}

Sub update_camera(oneX As Double, oneY As Double, oneZ As Double, oneDirection
    As Double, oneAngle As Double)
{
    My.X = oneX
    My.Y = oneY
    My.Z = oneZ
    My.Direction = oneDirection
    My.Angle = oneAngle
}
```

・・・コントロールパネルで指定した視点に切り替える

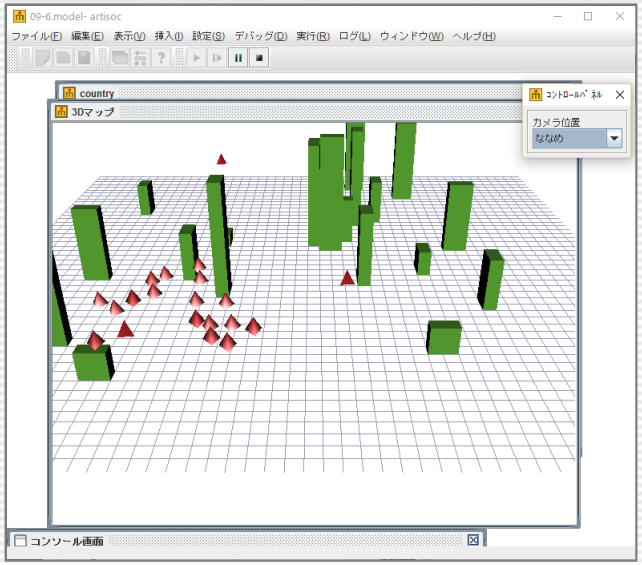
・・・カメラの状態を更新する

⑧-3 カメラ位置を指定する

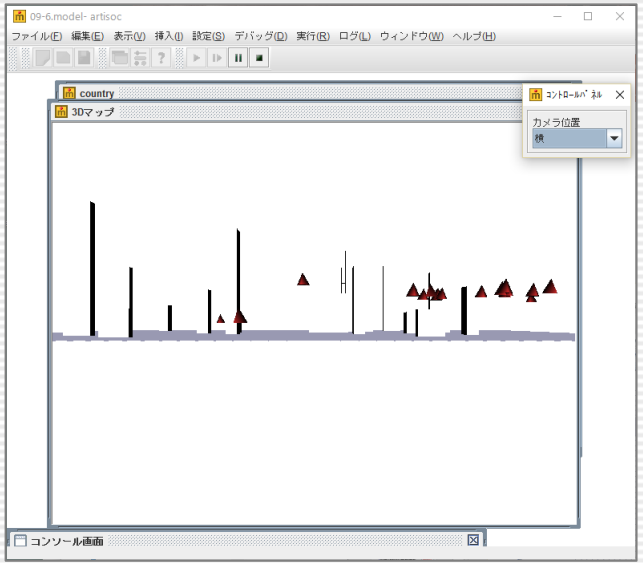
- シミュレーションを実行します。09-5.model
  - コントロールパネルで「真上」「ななめ」「横」の視点に切り替えられます。
  - update_camera関数に与えている引数を変更してみて、好きな視点を見つけてください。



真上



ななめ



横