変形ARマーカに対するAAE

ER17076

安井　理

AAEのトレーニング

- 現在までの問題
  ・自作の3Dモデルでは，トレーニングエラーを起こしてしまう

- エラー内容

UnicodeDecodeError: 'utf-8' codec can't decode byte 0x80 in position 260: invalid start byte

utfエラーは文字コードのエラーなのでモデルファイルに問題があると考えられる

- 解決方法として自作モデルと前回たサイトから拾ったネジのモデルを比較

トレーニング・推定時の入出力

- トレーニング
- 入力：3Dモデルから生成されたノイズの加わった画像
- 出力：オートエンコーダーよりエンコードされた物体画像

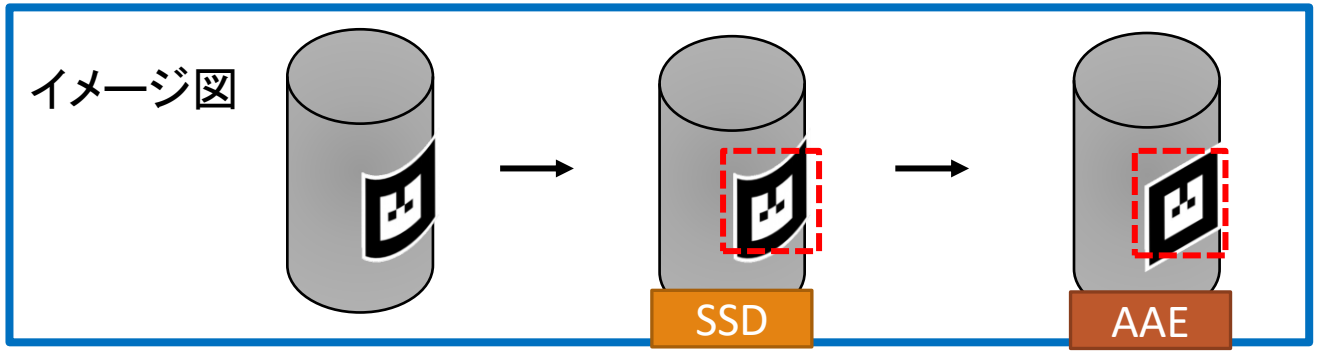
- テスト
- 入力：バウンディングボックスから得られた画像
- 出力：トレーニングで用意したエンコードされた画像で最も近い値の画像
  →この画像をバウンディングボックスに返す

トレーニング

- **トレーニングに3Dモデルを用意する必要性**

- OpenGLで3Dモデルを読み込み360°から見たそれぞれの画像を自動生成
- 実際のトレーニングの入力は，画像
- 姿勢推定時（テスト）の入力は画像→最終的にはwebカメラからの入力（チェック）

- **板状モデルを使う理由**
- 歪みのない平面状の画像をエンコードできるため出力をそのまま使うことができる



使用可能モデルとの比較

使えたモデル情報

ply
format ascii 1.0
comment VCGLIB generated
element vertex 5613
property float x
property float y
property float z
property float nx
property float ny
property float nz
property uchar red
property uchar green
property uchar blue
property uchar alpha
element face 2124
property list uchar int vertex_indices
end_header

自作の使えないモデル

ply
format ascii 1.0
comment Created by Open Asset Import Library - http://assimp.sf.net (v5.0.639693989)
element vertex 359
property float x
property float y
property float z
property float nx
property float ny
property float nz
property uchar red
property uchar green
property uchar blue
property uchar alpha
element face 129
property list uchar int vertex_index
end_header

モデル比較

- 黄色でマークしたプロパティ情報部分の違い
  - プロパティ名のint vertex_indexをint vertex_indicesに変更するとトレーニングが可能になる
  - トレーニングは可能だが，テスト時にエラーを起こす.（平面状物体）
  - indexとindicesは単数形と複数形の違い.トレーニングエラーを起こす理由は不明

トレーニングエラー

- 問題として考えた事
  - モデルサイズが小さすぎる事
    →blenderで前回使ったネジのモデルサイズは，縦横100mほどのサイズであった．(デフォルトは1m)
  - テクスチャが付けられている事
  - 平面状（XYの二次元）の物体が原因でないかという推測．

トレーニングエラー

- モデルの問題かどうかを確認として立方体のモデルを用意（blenderのデフォルトモデル）

- 立方体モデルでは[トレーニング・テスト]それぞれ正常に動作
  →サイズが小さいままだとテスト時に以下のエラーを起こす.

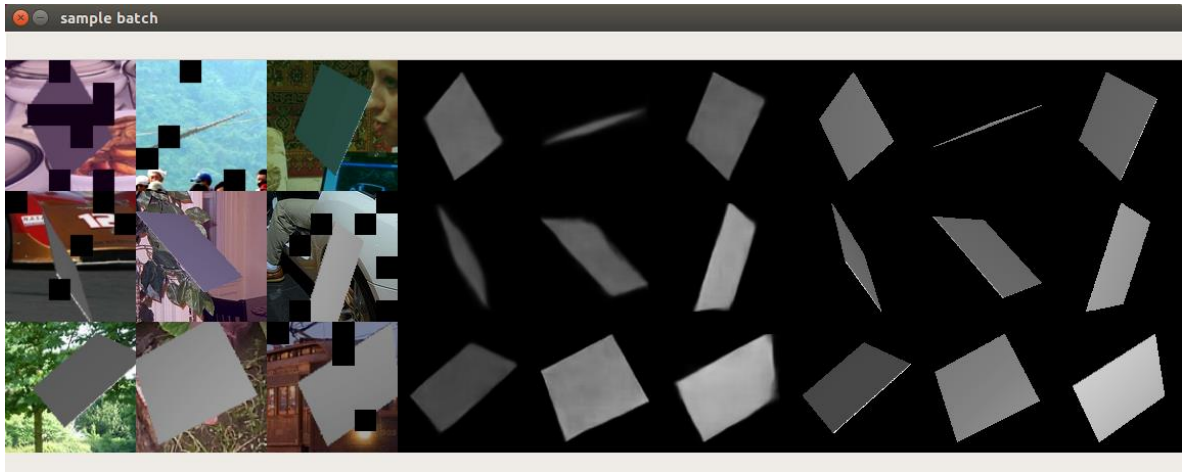
ValueError: zero-size array to reduction operation minimum which has no identity

・範囲にデータがないときに発生するエラー

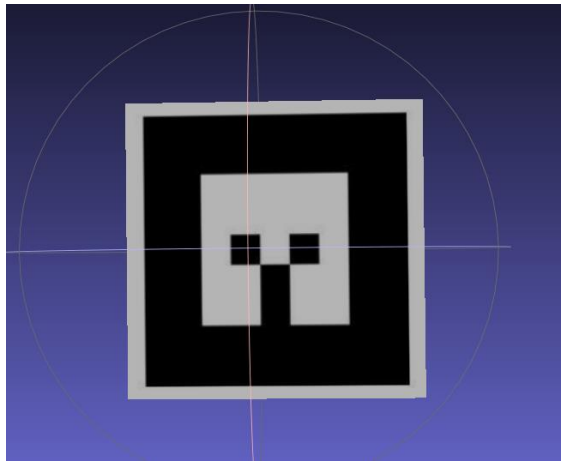
トレーニングエラー

- テクスチャを張り，再度トレーニング．
- トレーニングデータにテクスチャは反映されない．
- Meshlabで表示しているときにはテクスチャは反映されている．

→AAEでトレーニング画像を生成する際にテクスチャが読み込めない可能性



トレーニング時のエンコード



MeshLab

使用可能なモデル条件

- 平面状板　→　　使用不可　（2枚の板を結合すると使用可能）
- 立方体を薄くして板状にする　→　使用可能
- テクスチャの張り付けたモデル　→　使用可能（テクスチャ反映なし）

- Gazeboを使用しテクスチャが物体に反映されているかを確認

gazebo

- 目的:テクスチャが反映されているかを確認・SSDの学習で後に使用

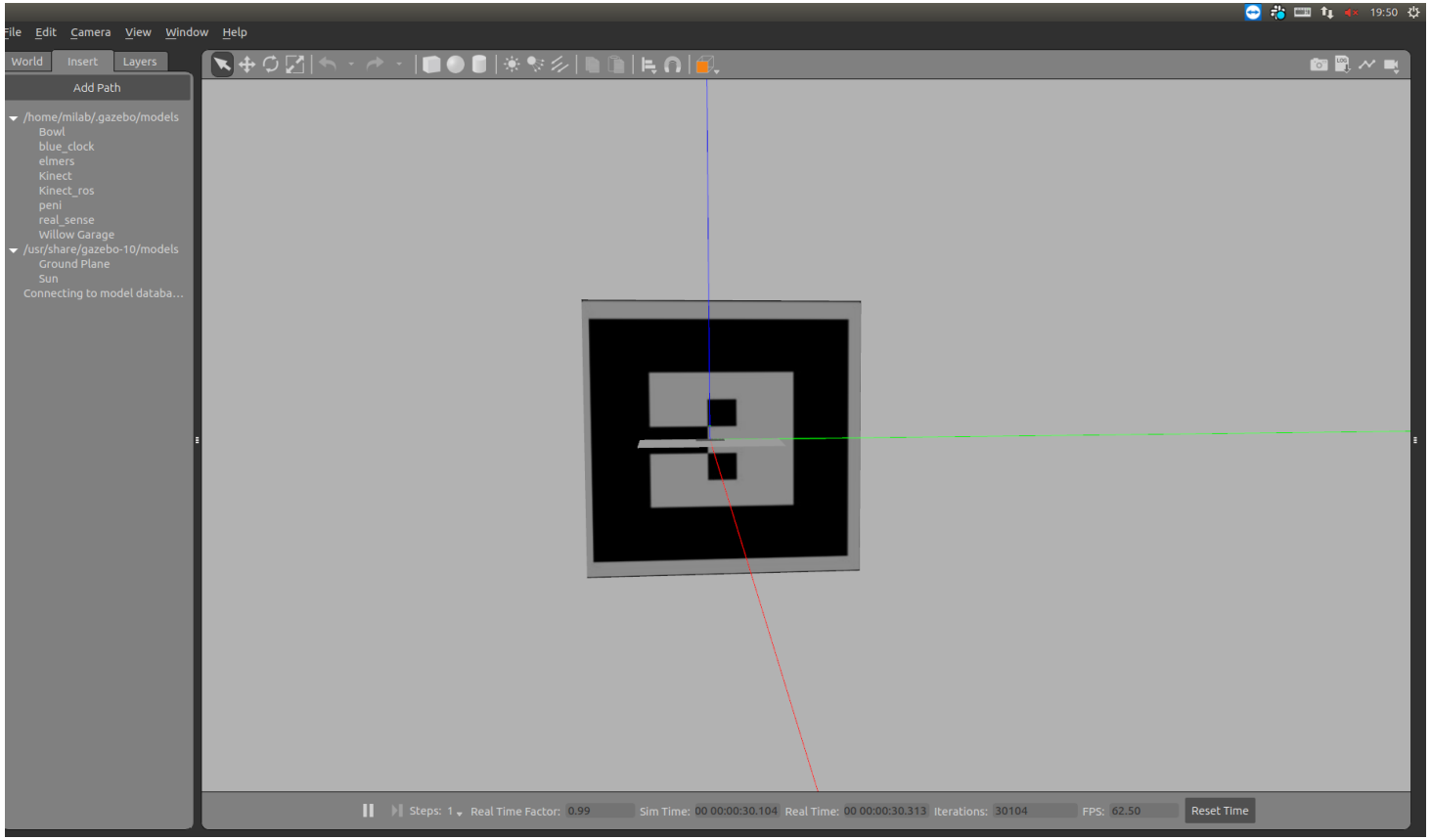
- １:隠しファイルを表示
- ２:ホーム→. gazebo→modelsの順番にファイルを入っていく
- ３:models内で自分のモデルを入れるファイルを作成
- ４:３で作成したファイルにコラッタファイル(.dae)を入れるファイルとconfigファイル・sdfファイルを作成
- ５:sdfファイル・configファイルにモデルファイルのpathを通す

- Gazeboを開いて
- Insertで３で作成した自分のファイルを選択するとgazebo上に表示される

gazebo

- Gazebo上の表示

テクスチャが反映されて
いることが確認できる



Modelの作成

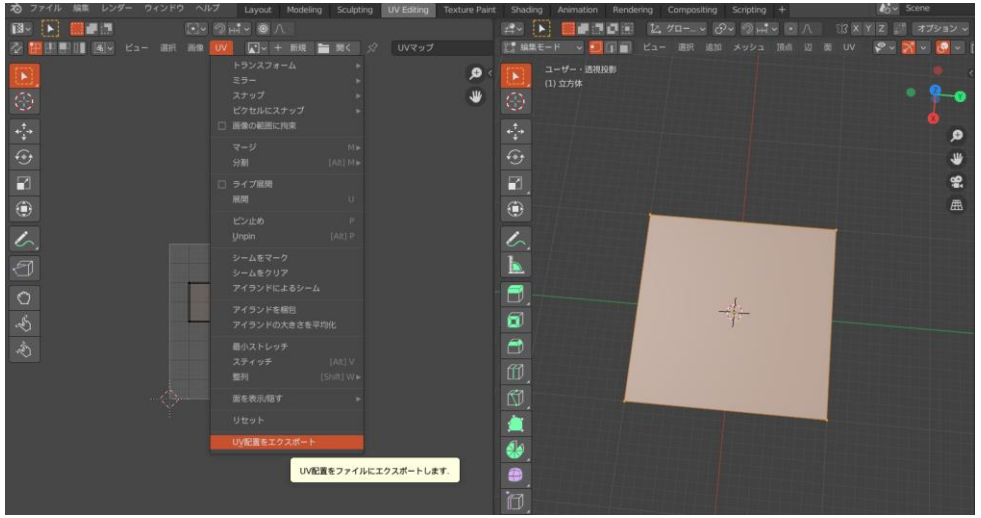
- Blenderを用いて作成

- １：blenderでデフォルトの立方体のサイズをｘ：100㎜，ｙ：100㎜，ｚ：1㎜，に変更
- ２：UVEditingに移動し左側の画面UV→UV配置をエクスポートを選択png画像を保存
- ３：Photoshopで画像png画像を表示しその中にARマーカの画像を張り付け，JPG形式で保存
- ４：blenderに戻り　マテリアル→ベースカラー→画像の選択（先ほど保存したJPG画像）
- ５：dae形式で保存→ウェブサイトを使用しdaeをply形式に変更
  - 直接ply形式で出力すると文字化けを起こしてしまう
- ６：plyに変更したファイルを開き
  16行目のproperty list uchar int vertex_indexをproperty list uchar int vertex_indicesに変更

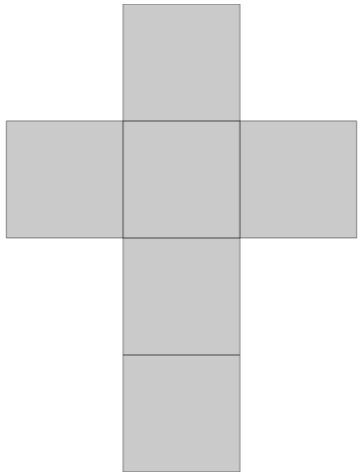
Modelの作成

- Blenderを用いて作成

- １:blenderでデフォルトの立方体のサイズをx:100㎜，y:100㎜，z:1㎜，に変更
- ２:UVEditingに移動し左側の画面UV→UV配置をエクスポートを選択png画像を保存



UV配置の保存

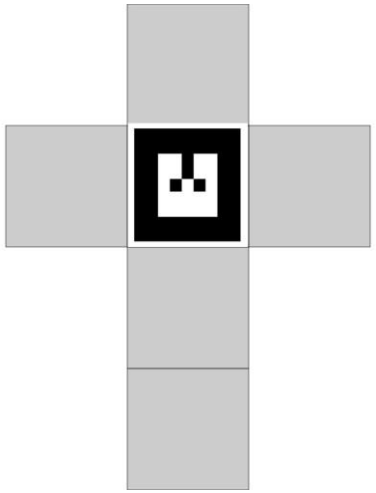


png画像

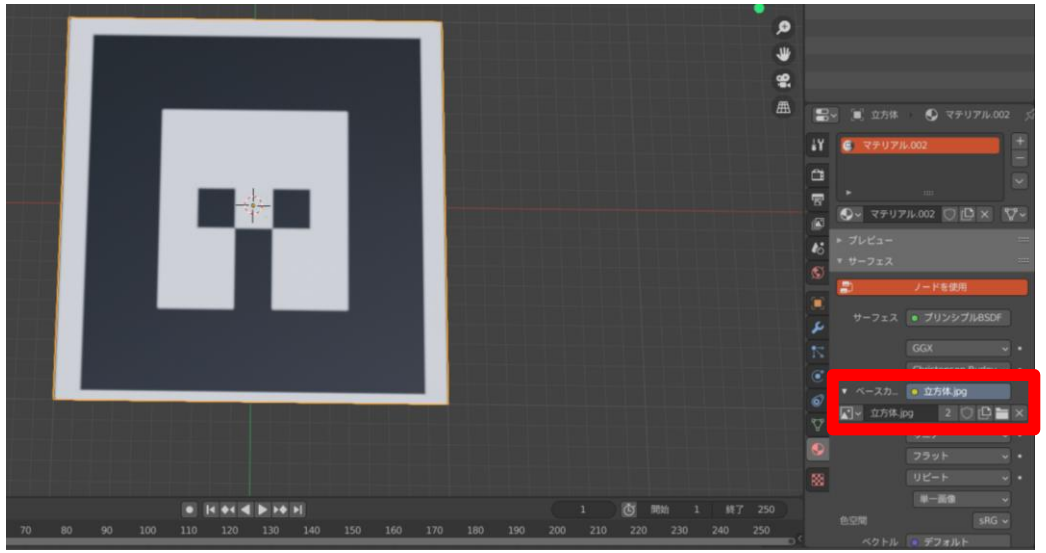
Modelの作成

- Blenderを用いて作成

- ３：Photoshopで画像png画像を表示しその中にARマーカの画像を張り付け，JPG形式で保存
- ４：blenderに戻り　マテリアル→ベースカラー→画像の選択（３で保存したJPG画像）



jpg画像



テクスチャの張り付け

Modelの作成

- Blenderを用いて作成

- １：blenderでデフォルトの立方体のサイズをx：100㎜，y：100㎜，z：1㎜，に変更
- ２：UVEditingに移動し左側の画面UV→UV配置をエクスポートを選択png画像を保存
- ３：Photoshopで画像png画像を表示しその中にARマーカの画像を張り付け，JPG形式で保存
- ４：blenderに戻り　マテリアル→ベースカラー→画像の選択（先ほど保存したJPG画像）

- ５：dae形式で保存→ウェブサイトを使用しdaeをply形式に変更
  ※blenderから直接ply形式で出力すると文字化けを起こしてしまう

- ６：plyに変更したファイルを開き
  16行目のproperty list uchar int vertex_indexをproperty list uchar int vertex_indicesに変更

次回までにやる事

- テクスチャが反映されない問題に対しての解決
- OpenGLについて調べ，テクスチャの対応なども調べる
- SSDからAAEへの入力について調査（榎本君と2人で行う）

参考文献

・6次元物体検出の論文

http://openaccess.thecvf.com/content_ECCV_2018/papers/Martin_Sundermeyer_Implicit_3D_Orientation_ECCV_2018_paper.pdf

・git-hub

https://github.com/DLR-RM/AugmentedAutoencoder#testing