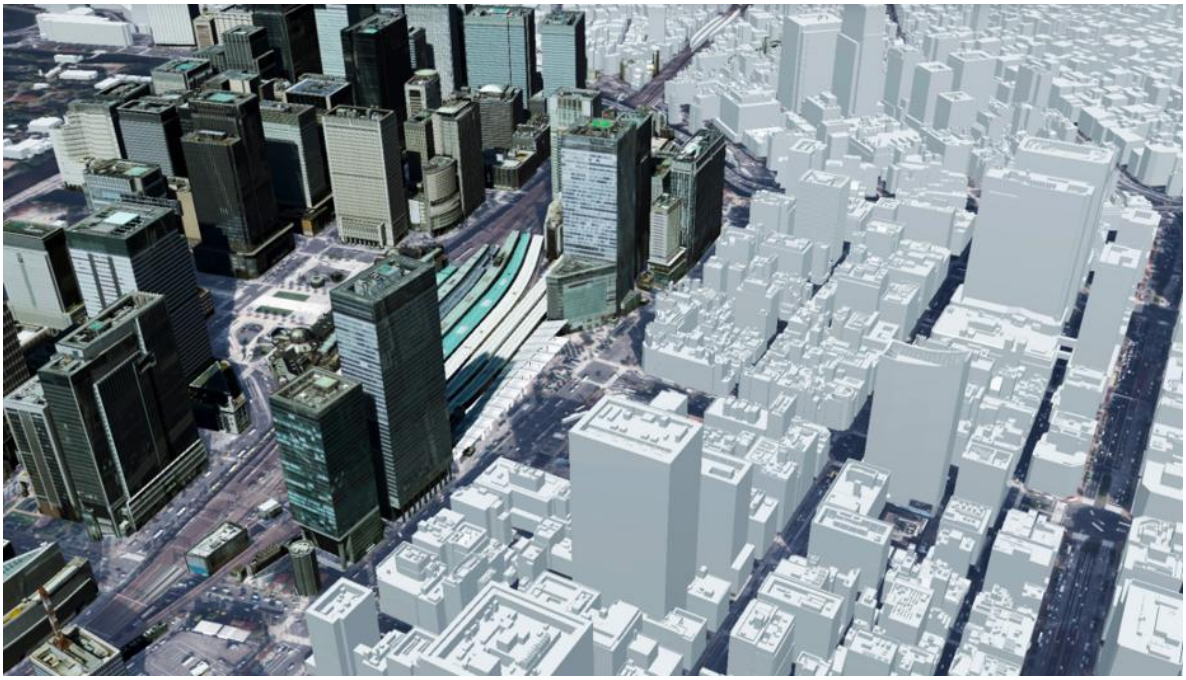
タイトル：第 1 回 写真からつくるデジタルツイン



About 3D City Model: https://www.mlit.go.jp/plateau/learning/

「デジタルツイン（Digital Twin）」という言葉を聞いたことがありますか？ 現在、注目を集める最新技術として「デジタルツイン」というものがあります。 コンピュータ(サイバー空間)で現実世界(フィジカル空間)に実在している街や建物を再現する技術のことです。「デジタルツイン=デジタルの双子」という言葉から、双子のように再現することで新しい産業が生まれるカギになるといわれています。

近年、AI や AR そして IoT などが発達したことでデジタルツインの実現に必要な技術やデータ収集方法が飛躍的に進化し、ミリ単位の精密さでデータが作成され可視化されています。

このシリーズはデジタルツイン入門として、関連技術やプログラムの紹介、都市などの広い空間で利活用されているインフラデータや現場での事例を紹介します。

「インフラ」とは：

「インフラストラクチャー(infrastructure)」の略。社会においての産業や生活の基盤となっている施設等です。例えば、道路、橋、鉄道、上・下水道の社会基盤や、公園、学校、病院、公共施設などの施設、そして、携帯電話網やインターネットなどの情報通信網等をいいます。

デジタルツインの例：

国土交通省と東京都が進めているプロジェクト

- Project PLATEAU [プラトー] - 国土交通省 **https://www.mlit.go.jp/plateau/**
  - 国土交通省が進めている、3D 都市モデル整備・活用・オープンデータ化のプロジェクト
- デジタルツイン実現プロジェクト - 東京都 **https://info.tokyo-digitaltwin.metro.tokyo.lg.jp/**
  - 東京都が進めている、デジタルツインを実現することで課題解決と都民の QOL 向上を目指しているプロジェクト

QOL: Quality of Life の略、生活の質、私たちが生きる上での満足度をあらわす指標

写真から 3D モデルを作成するフォトグラメトリ

まちのデジタルツインには 2 つのデータが存在します。

- 建物や道路などの現実に存在するモノ（=フィジカル空間）をデータ化する
- 現時点で存在しないモノを仮想空間（=空間）でモデル化（生成）する

デジタルツインでは、この 2 つのデータを 3DCG（3 次元コンピュータグラフィックス）技術を使用してサイバー空間に再現しています。

映画やゲームにある 3D(three-dimensional)映像に、現実空間の街やビルが再現されていることを想像するとイメージしやすいかと思います。

3D(3 次元)モデルの撮影・編集、そして現実空間の街や建物などをデータ化するためには、高価な機器や専門の知識が必要でした。しかし、スマートフォンやパソコンなどが進化したことにより、誰でも 3D モデルを手軽に作成することが出来るようになってきています。今回は、スマートフォンのアプリを使って 3D モデルを撮影して、撮影したデータを使った 3D モデルの生成方法を紹介します。

スマートフォンで 3D モデル生成

iPhone/iPad の一部の機種に LiDAR センサ※1 という赤外線等を使った深度センサーが搭載されたことにより、3D モデルを生成することが身近になりました。 スマートフォンやパソコンの性能が向上したことで、精度の高いモデルを作成できるようになりました。

スマートフォンで 3D モデルを生成する代表的な方法の 2 つに、LiDAR センサーを使用したものと、複数枚の写真から 3D モデルを生成するフォトグラメトリ（Photogrammetry）があります。

※1 LiDAR センサーの詳細については、次回以降に紹介をします。

LiDAR センサ

- レーザー光を照射し反射光や散乱光の反射時間を測定して、離れた物体の距離（深度=depth）を測る仕組み
- LiDAR センサは非常に高価で専用機器にしか搭載されていなかった

iPhone/iPad 搭載の LiDAR センサの特徴

- iPhone12/13Pro、iPad Pro(2020) に LiDAR センサが搭載された
- 計測範囲を最大 5m の距離に制限することでコストを抑えている
- 照射密度を低くしてコストを抑えている
- 小さいモノや複雑なモノを 3D モデル化するのは不得意
- 家の中の部屋くらいの広さまでの測定に向いている

フォトグラメトリ

- 複数枚の写真から 3D モデルを生成する
- 3D モデルを生成するには写真撮影時にコツが必要である
- 他のセンサからの情報（位置、角度、深度、距離等）を利用すること精度が向上する
- LiDAR センサ非搭載の iPhone や Android でも 3D モデルの生成が可能 (機種を選ばない)
- 写真の解像度などに影響する

フォトグラメトリアプリ

スマートフォンから３D モデルを生成するには専用のアプリを使用します。フォトグラメトリアプリは、無料版(機能制限あり)/有料版があり、アプリの種類も多くそれぞれに特徴があります。 初心者は WIDAR、Trnio、Polycam、Metascan あたりが扱いやすいと思います。

iPhone のフォトグラメトリアプリ

- WIDAR https://widar.io/
- Metascan https://metascan.ai/
- Trnio https://www.trnio.com/

Android のフォトグラメトリアプリ

- WIDAR https://widar.io/
- Polycam https://poly.cam/

フォトグラメトリアプリの比較や実際の撮影方法などは下記のブログに説明があります。こちらを参考にしてみてください

- 【初心者向け】iPhone 3D スキャンパーフェクトガイド
  https://note.com/iwamah1/n/n48a549845ae3

スマフォのアプリで *3D* モデル生成

スマフォアのアプリでフォトグラメトリから 3D モデル生成する場合は、撮影した画像や動画をサーバーにアップロードして、サーバーが特徴点抽出、マッチング、3D モデル生成することになります。

フォトグラメトリとは？

フォトグラメトリで 3D モデルを生成するには、被写体をいろんな角度や方向から撮影した写真の画像データを解析、統合して立体的な 3DCG モデル化します。通常の写真だけでも 3 D モデルの生成が可能ですが、3D モデルを生成するには最低でも約 20 枚～数十枚程度の画像データが必要となります。動画でも生成は可能です。

フォトグラメトリの生成方法

フォトグラメトリは、被写体は、カメラのフレーム内に被写体の全体が収まる形で、少しずつ角度を変えながら、写真が重なりあうように上下左右全体の写真を撮影したものが必要になります。

元の画像 *1*

元の画像 *2*

3D モデル生成画像

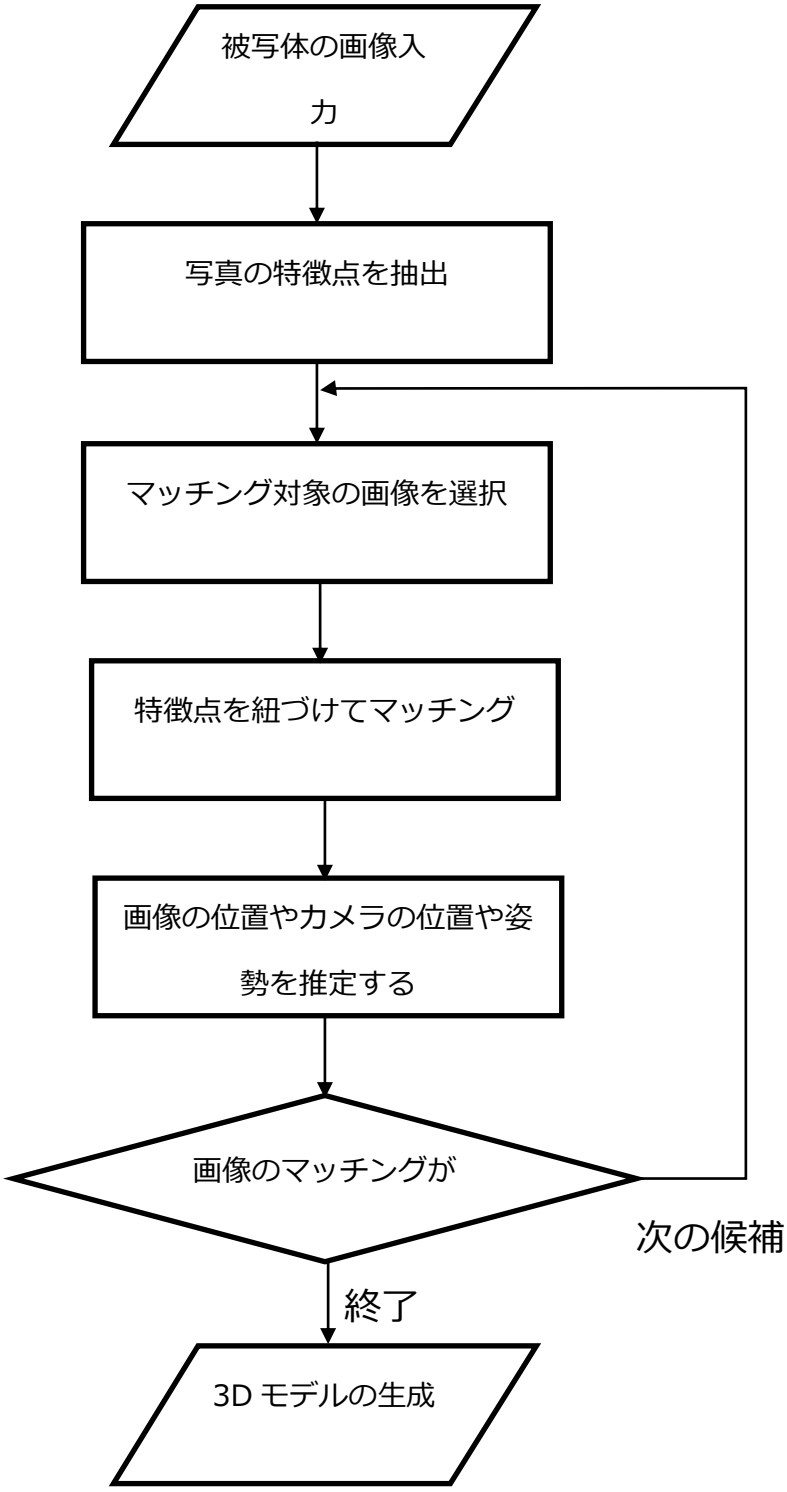
複数の写真から被写体の特徴点を抽出して、特徴点から被写体の位置、角度、方向を抽出して画像を合成して 3D モデルを生成していきます。

フォトグラメトリの被写体には向き・不向きがあります。

「表面に変化がある」「表面に模様がある」など特徴をとらえやすいものは、被写体に向いています。逆に「表面に変化が少ないもの」「反射して光ってしまうもの」「透明なもの」等は、被写体には向いていません。様々な角度からの写真を合成するので、静止してないものも被写体には向いていません。

フォトグラメトリの処理

フォトグラメトリの大まか処理の流れは下記となります。 3D モデルを作成するために、画像の特徴を分析して、画像の特徴情報を元に画像データから３D モデルを組み立てていきます。


被写体の画像入力
写真の特徴点を抽出
マッチング対象の画像を選択
特徴点を紐づけてマッチング
画像の位置やカメラの位置や姿勢を推定する
画像のマッチングが
次の候補
終了
3D モデルの生成

Python でフォトグラメトリ

フォトグラメトリの 3D モデル生成の基本部分は Python コードを使って紹介します。

Python の実行環境としては、Google Colaboratory を使用します。

画像解析にはオープンソースの画像処理・画像解析のためのライブラリ OpenCV https://opencv.org/を使用して画像の特徴点抽出とマッチングを試してみたいと思います。

OpenCV は、画像や動画の処理において汎用性が高く Python のライブラリとしては人気が高いライブラリです。OpenCV を使用することで、画像や動画の物体の位置情報やパターン、動きの識別ができるようになり、応用範囲も広いです。

今回は下記のサイトを参考にしています。

- OpenCV: 特徴点抽出とマッチング https://ohke.hateblo.jp/entry/2019/08/03/235500
- OpenCV で特徴量マッチング(AKAZE, KNN) サンプルコード付 https://miyashinblog.com/opencvakaze-knn/
- Google Colaboratory で OpenCV をためす https://qiita.com/toyohisa/items/cbcdb0cb20265c4168c7

OpenCV: 特徴点抽出とマッチング

Google Colaboratory で画像データを処理するためには、画像データをアップロードする必要があります。実行の準備として下記のいずれかの方法でコードから画像データを読み込む準備をしてください

画像データを直接アップロードする

画像データを直接アップロードする場合は下記の方法でアップロードをします。

```
from google.colab import files # colaboratory のライブラリからファイル操作を使用する。
f = files.upload()
```

アップロードした画像は、Google Colaboratory のセッションが切れたり再起動したりした場合、削除されてしまうので、セッションが切れたときは再度アップロードする必要があります。

Google ドライブをマウントして画像データを読み込む

Google Colaboratory は、Google ドライブのファイルを読み込むことが可能です。 Google ドライブのファイルを読み込めるようにするには下記の方法で Google ドライブをマウントします。

```
from google.colab import drive
drive.mount('/content/gdrive')
```

画像データを表示

画像データのテストデータの例

画像データを表示してみます

```
import cv2
import matplotlib.pyplot as plt

# サンプル画像（/content/gdrive/MyDrive は Google ドライブのパス）
```

```
file1 = "/content/gdrive/MyDrive/001.png"  # 画像 1
file2 = "/content/gdrive/MyDrive/002.png"  # 画像 2

# 画像のロード
img1 = cv2.imread(file1)
img2 = cv2.imread(file2)

# 画像の表示関数
def display(img):
  show_img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB) # 表示のため BGR を RGB に変換する。
  plt.axis('off')
  plt.imshow(show_img) # matplot.lib を用いて読み込んだ画像を表示します。

# 画像の表示
display(img1)
```

特徴点抽出

OpenCV には、画像の特徴点抽出アルゴリズムとして数種類のアルゴリズムが用意されています。その中から、AKAZE(Accelerated KAZE)という回転や拡大，輝度変化に対応できる特徴点抽出アルゴリズムを使用して特徴点を抽出してみます。 A-KAZE は、KAZE というア

ルゴリズムを高速化して A-KAZE(Acclerated KAZE)となったものです。他のアルゴリズムには特許権があり使用には注意や使用申請が必要ですが、AKAZE は商用/非商用を問わず利用可能です。

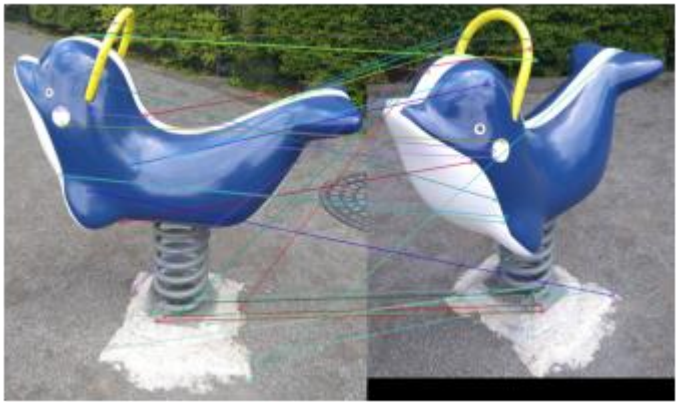
```
# グレースケール変換
gray1 = cv2.cvtColor(img1,cv2.COLOR_BGR2GRAY) # 画像 1
gray2 = cv2.cvtColor(img2,cv2.COLOR_BGR2GRAY) # 画像 2

#  AKAZE 検出器の生成
akaze = cv2.AKAZE_create()

# gray1 に AKAZE を適用、特徴点を検出
key_point1, descriptions1 = akaze.detectAndCompute(gray1,None)
# gray2 に AKAZE を適用、特徴点を検出
key_point2, descriptions2 = akaze.detectAndCompute(gray2,None)

# キーポイントの表示
extraceted_img1 = cv2.drawKeypoints(gray1, key_point1, None, flags=4)
extraceted_img2 = cv2.drawKeypoints(gray2, key_point2, None, flags=4)
display(extraceted_img2)
```

グレースケール変換して、特徴点を表示しています。

マッチング

抽出した各画像の特徴点のマッチングをします。マッチング結果から画像ごとの位置や向きを推定し特徴点同士を結びつけます。

```
def match(img1, img2):
    # 各画像の特徴点を取る
    key_point1, descriptions1 = akaze.detectAndCompute(img1,None)
    key_point2, descriptions2 = akaze.detectAndCompute(img2,None)

    # BFMatcher オブジェクトの生成
    bf_matcher = cv2.BFMatcher(cv2.NORM_HAMMING, crossCheck=True)
    # 2 つの特徴点をマッチさせる
    matches = bf_matcher.match(descriptions1, descriptions2)

    # matches を descriptors の似ている順にソートする
    matches = sorted(matches, key = lambda x:x.distance)
    matches_length = len(matches) # マッチした本数

    # 特徴点同士を線でつなぐつなぐ画像を作成（線はマッチ順の先頭 30 本まで表示）
    match_img = cv2.drawMatches(img1, key_point1, img2, key_point2, match
es[:30], None, flags = cv2.DrawMatchesFlags_NOT_DRAW_SINGLE_POINTS)

    return match_img, matches_length

# 画像のロード
match_img, matches_length= match(img1, img2)
display(match_img)
print(f"matches_length={matches_length}")
```



マッチした線が見やすいように 30 本までを表示しています（実際の結果では 174 本マッチしています）

3D モデル生成

3D モデル生成には、Meshroom https://github.com/alicevision/meshroom/releases というパソコン向けのオープンソースのアプリケーションを使用します。

Meshroom は、AliceVision https://alicevision.org/をベースとして作られたフォトグラメトリ用のアプリケーションで、無料で使用でき、初心者の方でも比較的扱いやすいアプリです。

GPU 環境の準備

Google Colaboratory メニューから GPU を有効にして GPU 実行の設定をします。

「メニュー選択」→「ランタイム」→「セッションのタイプを変更」→「ハードウェアアクセラレータ」→「GPU を設定」

```
import tensorflow as tf
device_name = tf.test.gpu_device_name()
if device_name != '/device:GPU:0':
  raise SystemError('GPU device not found')
print('Found GPU at: {}'.format(device_name))
```

Google ドライブをマウント

画像データの Google ドライブのコピーし Google ドライブをマウントします。

```
from google.colab import drive
drive.mount('/content/drive')
```

Meshroom をダウンロード

Meshroom をダウンロードして、実行できるようにコードを展開します。

```
# get Meshroom
!wget -N https://github.com/alicevision/meshroom/releases/download/v2021.1.0/Meshroom-2021.1.0-linux-cuda10.tar.gz
!mkdir meshroom
!tar xzf Meshroom-2021.1.0-linux-cuda10.tar.gz -C ./meshroom
```

計算結果の出力先を作成

Meshroom の出力先ディレクトリを作成します

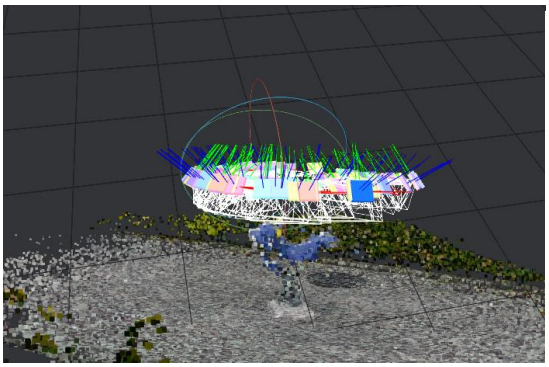
```
!mkdir ./object_out
```

Meshroom を実行

Meshroom を実行します。 画像データはディレクトリ配下のすべての画像が対象になります。

- 画像データ: ./drive/MyDrive/dataset
- 出力先： ./object_out

```
!./meshroom/Meshroom-2021.1.0-av2.4.0-centos7-cuda10.2/meshroom_batch --input ./drive/MyDrive/dataset --output ./object_out
```

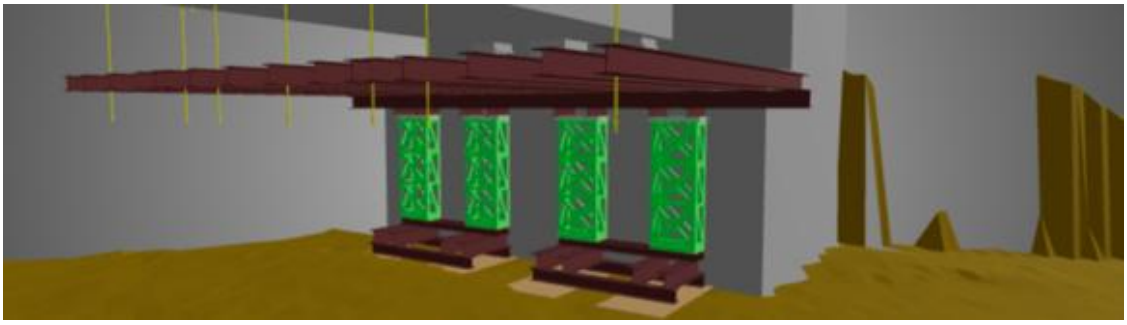


参考

Meshroom に使い方は下記が参考になります

- 【フォトグラメトリ】Meshroom を使って 3D モデルを作る
  https://styly.cc/ja/tips/photogrammetry-rinmizouchi-meshroom/

事例：橋梁の点検



このフォトグラメトリの手法は、身近な社会でも用いられ、街の安心・安全を維持するために使われています。道路の橋梁を点検は、従来は人が行っていましたが、最近ではドローンを用いた点検が行われています。人による点検では、点検者が橋梁の細部まで見に行って、ひび割れやサビなどの損傷がないか確認を行っています。橋の柱や下面まで人が、労力を要し危険も多いため、カメラを搭載したドローンで橋梁を撮影し点検する手法が最近用いられるようになりました。200m程度の橋梁をドローン点検で撮影すると、1 回に数百～数千の画像を取得することになります。取得した膨大な写真の撮影位置を割り出し、

損傷位置を解析するためにフォトグラメトリを用いて、点検の３D モデルと画像の撮影位置から、損傷の位置や程度を把握します。 3D の橋梁モデルは、損傷位置や周辺環境を確認するためにも役立ち、維持管理や損傷の補修計画を策定するために、工事関係者と情報共有をスムーズに進めることができます。

事例：みんなの首里城デジタル復元プロジェクト

世界文化遺産に登録され、国指定史跡でもある沖縄県の首里城が 2019 年 10 月 31 日の火災により正殿などが焼失しました。首里城が焼失したのは歴史上 5 度目と言われていて、現在、正殿等の復元・復興を目指しています。

実際の正殿などの復元・復興とは別に、コンピュータを使って首里城をデジタルで復元しようと「みんなの首里城デジタル復元プロジェクト」というプロジェクトも立ち上がりました。 このプロジェクトは、焼失前に一般の方が撮影した首里城の写真やビデオを収集し、焼失前の在りし日の首里城を復元するものです。 復元プロジェクトのプロジェクトメンバーはボランティアで集まった研究者や学生やエンジニアがメンバーで、一般の方が撮影した写真を広く募集し画像データを収集するところがこのプロジェクトの特徴です。このプロジェクトにフォトグラメトリ技術の 1 つである Structure from Motion（SfM）が使われています。

- 首里城復興へのあゆみ https://oki-park.jp/shurijo/fukkou/
- みんなの首里城デジタル復元プロジェクト
  - 公式サイト https://www.our-shurijo.org/
  - YouTube https://www.youtube.com/watch?v=Ac4HM42DG5Q
  - Twitter https://twitter.com/our_shurijo
  - 情報処理 Vol.61 No.2 Feb. 2020 特別解説 OUR Shurijo みんなの首里城デジタル復元プロジェクト https://www.ipsj.or.jp/magazine/9faeag000000zx98-att/Shurijo.pdf