3S-07

# 軍艦島モニタリングに向けた首振りカメラによる映像取得方式の検討

守屋広汰 † 小寺志保 † 倉田成人 ‡ 濱本卓司 ‡† 猿渡俊介 †
† 静岡大学 ‡ 筑波技術大学 ‡† 東京都市大学

## 1 はじめに

軍艦島は，長崎県にある無人島であり，現在も経年劣化による建物の複雑な崩壊現象が発生している．筆者らは，軍艦島において崩壊中の建物の映像や音声，加速度といったデータを収集することで，建築構造解析に貢献することを目指して軍艦島モニタリングプロジェクト [1] を進めている．本稿では，軍艦島モニタリングにおいて，首振りカメラを用いて効率的に映像を撮影する方法について提案する．シミュレーションと実測によって性能を検証し，提案手法が少ない計算コストで最適な手法に漸近する性能を達成できることを示す．

## 2 軍艦島モニタリングにおける映像処理

図 1 に軍艦島モニタリングのシステム構成を示す．Panasonic BB-SW175A の首振り機能を利用して軍艦島内の広い範囲の映像を撮影する．建物の経年劣化をモニタリングするために，島内で最も高い場所に位置する 3 号棟より首振りカメラを用いて映像を取得している．撮影された映像はノート PC に転送された後に，センサデータと共に 25 GHz 帯の無線通信によって本土へと伝送される．

図 1 の軍艦島モニタリングシステムで構造建築物の映像収集するために，筆者らは，データサイズを削減しつつ高い映像品質を実現するパノラマとタイムラプスを利用した時空間圧縮方式の検討を進めている [2]．パノラマとタイムラプスの 2 つを組み合わせることにより，時間方向と空間方向に対する相関を利用した映像の圧縮が可能となる．

## 3 首振りカメラを用いた映像撮影における課題

2 節で述べた時空間圧縮方式を用いることで，取得した映像は映像品質を維持したまま，データサイズを削減することができる．しかしながら，軍艦島モニタリングシステムにおける映像処理では，映像撮影方法に関しても課題が存在する．図 1 に示した通り軍艦島モニタリングでは首振りカメラを用いて映像を撮影するため，カメラをどう動かすかによって，取得できる映像や 1 度の撮影にかかる時間が異なる．首振りカメラを効率よく動かして映像を撮影するためには，以下の 2 つの要件を同時に満たす必要がある．

1 つ目は，首振りカメラで撮影できる範囲は全て撮影することである．建物の崩壊現象を正確に記録するためには，広範囲にわたって映像を撮影する必要がある．2 つ目は，首振りカメラが移動する経路を短くすることである．軍艦島モニタリングは太陽光発電を用いて電力を供給しているため，カメラを動かすための電力はできる限り少ない方が望ましい．また，首振りカメラはすべての撮影範囲を 1 度撮影した後，撮影を開始した点に戻って撮影を再開するため，撮影終了地点から撮影開始地点へ戻る経路も含めた巡回する経路を検討する必要がある．

図 2 に，撮影範囲が $4 \times 3$ である場合の首振りカメラ移動経路の一例を示す．撮影範囲の左上を 0 として順に番号を振ると，図 2 の経路は 0，1，2，3，4，5，6，7，8，9，10，11 と移動した後，また 0 に戻る経路になる．図 2 の経路ではすべての範囲を撮影できているが，3 から 4 に移動する経路や 11 から 0 に戻る経路が長くなり，無駄な経路が発生することがわかる．そこで本研究では，首振りカメラが最も効率よく巡回するためのアルゴリズムを検討する．

首振りカメラが最も効率よく巡回するためのアルゴリズムは，巡回セールスマン問題と捉えることができる．巡回セールスマン問題とは，すべての点を 1 回ずつ巡り，出発点に戻ってくるまでの最短経路を探索する問題である．各点間の経路にはそれぞれ距離などの移動コストがあるため，総コストが最小となる経路を計算する．一方，首振りカメラの動きに関する条件は，指定された角度への移動

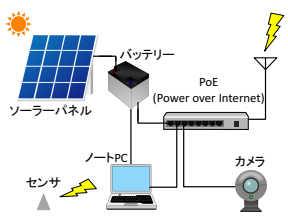

図 1: 軍艦島モニタリングシステム

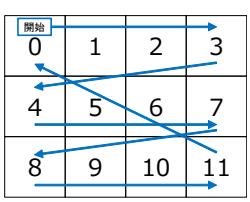

図 2: カメラの巡回例

**Algorithm 1** Select the pass with low cost

```
T_LNG ← calc_time (from_LNG (H, W))
T_LAT ← calc_time (from_LAT (H, W))
if T_LNG < T_LAT then
    Select the pass from LAT
else
    Select the pass from LNG
end if
```

表 1: *Algorithm* 1 で使用する変数，関数

| 変数，関数 | 説明 |
|---|---|
| $H$ | 縦の点の数 |
| $W$ | 横の点の数 |
| $T_{\mathrm{LNG}}$ | 縦方向からスタートする経路の巡回時間 |
| $T_{\mathrm{LAT}}$ | 横方向からスタートする経路の巡回時間 |
| calc_time $(a)$ | 経路 $a$ の巡回時間を計算する関数 |
| from_LNG $(h, w)$ | $h \times w$ の巡回経路を選択する関数 (縦方向からスタート) |
| from_LAT $(h, w)$ | $h \times w$ の巡回経路を選択する関数 (横方向からスタート) |

を繰り返すことと，上下左右，斜めへの移動が可能であることである．首振りカメラが指定された角度を頂点，別の角度への移動を頂点間の辺とすると，首振りカメラが巡回する動作はすべての頂点を移動して始点に戻る動作だと捉えられる．首振りカメラの角度変更にかかる時間を各辺におけるコストとすると，効率の良い巡回経路とは総コストが最小になる経路となる．すなわち，首振りカメラが最も効率よく巡回するためのアルゴリズムは，始点を出発してから始点へ戻ってくるまでのコストを最小化する巡回セールスマン問題として解くことができる．しかしながら，巡回セールスマン問題では，頂点数 $V$，辺の数 $E$ の場合，計算量が $O(V!)$ となり，計算コストが大きくなる．動的計画法を用いたとしても計算量は $O(V^2 2^V)$ である．計算量から巡回する頂点数が少ない場合には動的計画法で高速に厳密解を計算することが可能である．

## 4 提案手法

3 節で述べた首振りカメラ巡回に関する問題を解決するために，少ない計算コストで首振りカメラを巡回させるためのアルゴリズムを提案する．

*Algorithm* 1 に首振りカメラの巡回経路を決定するアルゴリズムを示す．*Algorithm* 1 では，縦方向からスタートする場合と横方向からスタートする場合の経路を比較し，2 つの経路からコストの少ない経路を選択する．首振りカメラが少ないコストで巡回するためには，できるだけ同じ方向に長く進む経路を選択する必要がある．左上からスタートする場合，同じ方向に長く進む経路は，左上から左下へ縦方向に移動する経路と左上から右上へ横方向に移動する経路が挙げられる．表 1 に *Algorithm* 1 で使用する変数と関数を示す．首振りカメラが巡回する範囲のうち，$H$ は縦方向の点の数，$W$ は横方向の点の数を表す変数である．$T_{\mathrm{LNG}}$ と $T_{\mathrm{LAT}}$ はそれぞれ，縦方向からスタートする場合と横方向からスタートする場合の巡回時間であり，経路のコストになる．calc_time $(a)$ は経路 $a$ の巡回時間を計算する関数，from_LNG $(h, w)$，from_LAT $(h, w)$ は $h \times w$ の範囲における巡回経路を算出する関数である．from_LNG $(h, w)$ は縦方向からスタートする経路，from_LAT $(h, w)$ は横方向からスタートする経路を算出する．from_LNG $(h, w)$ と from_LAT $(h, w)$ の詳細は後述する．*Algorithm* 1 では，まず，縦方向からスタートする経路と横方向からスタートする経路の巡回時間を算出する．次に，巡回時間に従って経路を決定する．$T_{\mathrm{LAT}}$ の方が小さい場合，横方向の移動からスタートする経路を採用し，$T_{\mathrm{LNG}}$ の方が小さい場合，縦方向の移動からスタートする経路を採用する．

*Algorithm* 2 に from_LNG $(h, w)$ のアルゴリズムを示す．from_LAT $(h, w)$ のアルゴリズムは *Algorithm* 2 の $i$ と $j$，$H$ と $W$ を入れ替えたものと同じになる．

表 2 に *Algorithm* 2 で使用する変数と関数を示す．$i$ は縦の位置，$j$ は横の位置を表す変数，move_perlength $(b, c)$ は現在の点 $b$ を $c-1$ だけ移動させる関数，move_perone $(d)$ は 1 つ分 $d$ を移動

**Algorithm 2** from_LNG $(h, w)$

**Input:** $H, W$

```
 1: i ← 0
 2: j ← 0
 3: i ← H − 1
 4: while i = 1 do
 5:     move_perlength(j, W − 1)
 6:     i ← i − 1
 7: end while
 8: if j = 1 then
 9:     move_perlength(j, W − 1)
10:     i ← i − 1
11:     j ← 0
12: else
13:     while j = 1 do
14:         move_perone(i)
15:         j ← j − 1
16:     end while
17: end if
18: if i = 0 then
19:     i ← 1
20: else
21:     i ← 0
22: end if
23: j ← 0
```

表 2: *Algorithm* 2 で使用する変数，関数

| 変数，関数 | 説明 |
|---|---|
| $i$ | 縦の位置を表す変数 |
| $j$ | 横の位置を表す変数 |
| move_perlength $(b, c)$ | 現在の点 $b$ を $c-1$ だけ移動させる関数 |
| move_perone $(d)$ | 現在の点 $d$ を 1 つ分移動させる関数 |

させる関数である．move_perlength $(b, c)$ では，現在の点を $b$，行または列の端の点 $c$ をとして，現在の点 $b$ が端の点 $c$ にいない場合は点 $c$ まで進み，点 $b$ が点 $c$ にいる場合は点 $c$ から 1 行目まで進む．*Algorithm* 2 では，まず始点 $(0, 0)$ から点を左下へ移動させる．次に，巡回していない点が残り 2 行になるまで行の点を全て通過しながら上へ移動する．$H$ が偶数の場合，巡回していない点が残り 2 行になった時点で点が $(1, 1)$ にあるため，これまでと同様に全ての点を巡回しながら始点 $(0, 0)$ まで移動する．一方，$H$ が奇数の場合，巡回していない点が残り 2 行になった時点で点が右端にあるため，1 列ずつ縦に移動しながら全ての点を移動し，始点 $(0, 0)$ に戻る．1 列ずつ移動している途中で経路が閉じた場合のみ，始点 $(0, 0)$ へ斜めに移動する．

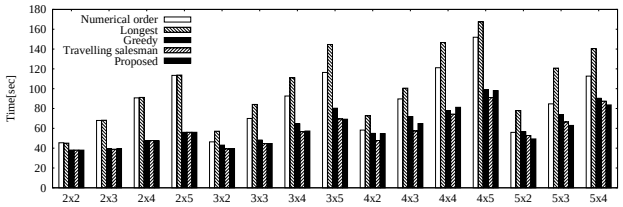


図 3: 移動時間

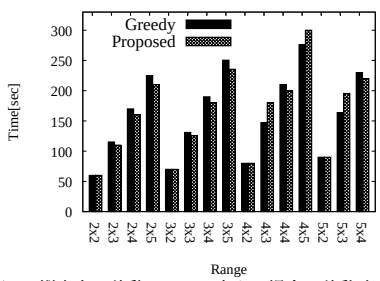


図 4: 縦方向の移動コストが大きい場合の移動時間

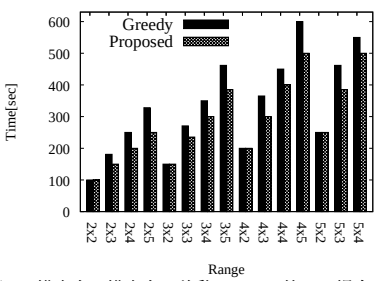


図 5: 横方向と横方向の移動コストが等しい場合の移動時間

## 5 性能評価

まず，4 節で述べた経路選択アルゴリズムを首振りカメラ上に実装し，選択した経路の巡回時間を評価した．首振りカメラには Panasonic BB-SW175A を用いた．首振りカメラに対してサーバから角度を指定する．サーバは dynabook R732/E26GB(Intel Core i5-3230M) 上で動作させた．首振りカメラの任意の角度間の移動時間は常に変化しないと考えられるため，あらかじめ指定された角度間の移動にかかる時間を算出しておく．指定した角度が $n$ 個あった場合，全頂点間である $n^2$ 個分の移動時間を算出する．巡回する範囲は 2×2 から 5×4 までとし，左上から順に 0，1，…，N と番号を振る．始点からすべての頂点を通ってまた始点に戻るまでの移動時間を計測した．提案手法の性能を相対的に評価するため，0，1，…，N の順に点を移動する numerical order，最も時間のかかる経路を選択する longest，貪欲法によって総コストは考慮せず，現在の地点から最もコストの低い経路を選択する greedy，巡回セールスマン問題のアプローチで経路を選択する traveling salesman と，提案手法 proposed を比較した．なお，traveling salesman は最もコストが低くなるように経路を選択するため，最短経路の基準となる．

図 3 に，各範囲における首振りカメラの移動時間を示す．縦軸が首振りカメラの移動時間 [sec] である．図 3 より，どの範囲においても proposed による経路が traveling salesman による経路に近い移動時間を実現していることがわかる．

移動時間の評価で用いた首振りカメラは，縦方向への移動コストと横方向への移動コストがほとんど同じであった．そこで，縦方向と横方向の移動コストが異なる場合の経路の移動時間を比較した．シミュレーションを用いて，縦方向への移動コストを 25，横方向への移動コストを 5 とした場合と，縦方向への移動コストを 5，横方向への移動コストを 5 とした場合の greedy と proposed の経路の移動時間を計測した．図 4 に縦方向への移動コストを 25，横方向への移動コストを 5 とした場合の移動時間，図 5 に縦方向への移動コストを 25，横方向への移動コストを 25 とした場合の移動時間を示す．縦軸は移動時間 [sec]，横軸は頂点の個数である．図 4，図 5 から，proposed と greedy の差が殆ど無いことが分かる．

次に，アルゴリズムの効率について比較するために，シミュレーションによって経路の計算時間を求めた．移動範囲を N × N とし，N を 2 から 100 まで変化させた場合の greedy，travelling salesman，proposed の計算時間を算出した．巡回セールスマン問題を動的計画法を用いて解いた場合，メモリ空間が $O(N2^N)$ 必要になる．使用した評価環境ではメモリに限りがあるため，travelling salesman は N が 4 までの計算時間を結果とした．図 6 に $2 \leq N \leq 100$ までの greedy，traveling salesman，proposed の計算時間を示す．グラフの縦軸が計算時間 [ms]，横軸が N である．N=5 の場合，traveling salesman によるアプローチでは計測ができなかったため便宜的に $10^4$[ms] にプロットしている．図 6 から，greedy，travelling salesman は N が増加すると計算時間も増加するが，proposed の計算時間はほとんど増加しないことがわかる．

## 6 おわりに

本稿では，軍艦島モニタリングに向けた映像撮影方式について検討した．首振りカメラを効率よく動かすことによって，広範囲の映像を短時間で取得することができる．現在，提案手法を実際の軍艦島モニタリングに適応させることを進めている．

### 参考文献

[1] 軍艦島モニタリングプロジェクト, http://sarulab.inf.shizuoka.ac.jp/battleship
[2] 小寺 志保, 倉田 成人, 濱本 卓司, 渡辺 尚, 猿渡 俊介, ”軍艦島モニタリングに向けた映像処理方式の提案”, 電子情報通信学会ソサイエティ大会, 2015 年

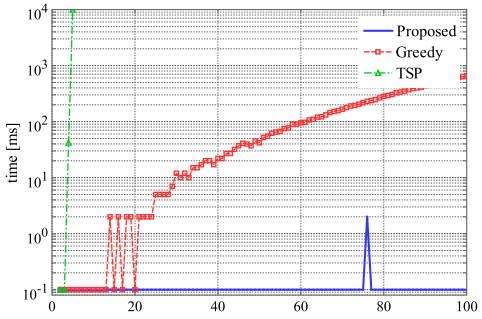


図 6: $2 \leq N \leq 100$ における計算時間