Optimization of Aerial Surveys using an Algorithm Inspired in Musicians Improvisation

João Valente
joao.valente@upm.es
Index

1. Introduction
2. Problematic
3. Harmony Search algorithm
4. The m-CPP algorithm
5. Results achieved
6. Conclusions
Introduction

• **Goal:**
  - Compute trajectories for a fleet of mini aerial vehicles shipped with a digital camera subject to a set of restrictions
  - Mosaicking

• **Applications**
  - Monitoring and inspections of Critical infrastructures
  - Precision agriculture

• **Projects:**
  - ROTOS (Multi-Robot System for Large Outdoor Infrastructures Protection. DPI 2010-17998)
  - RHEA (Robot Fleets for Highly Effective Agriculture and Forestry Management. NMP-CP-IP 245986-2)
Problematic

Full coverage trajectories
Harmony Search algorithm (I)

• Basic concepts
  • Soft computing, Meta-heuristic approach
  • Inspired by the improvisation process of musicians

• Methodology
  • Step 1: Initialization of the optimization problem
  • Step 2: Initialization of the harmony memory (HM)
  • Step 3: Improvisation a New Harmony from the HM set
  • Step 4: Updating HM
  • Step 5: Repeat steps 3 and 4 until the end criterion is satisfied

Harmony Search algorithm (II)

• Step 1: Initialization of the optimization problem

Minimize $F(x)$ subject to $x_i \in X_i$, $i = 1, 2, ..., N$

Where:

- $F(x)$: Objective function
- $x$: Set of each design variable ($x_i$)
- $X_i$: Set of the possible range of values for each design variable ($a < x_i < b$)
- $N$: Number of design variables
Harmony Search algorithm (III)

- **Step 2**: Initialization of the harmony memory (HM)
  - Generate random vectors
  - HMS: Harmony Memory Size

\[
HM = \begin{bmatrix}
X_1^1 & \ldots & X_N^1 & J(X_1^1) \\
\vdots & \ddots & \vdots & \vdots \\
X_1^{\text{HMS}} & \ldots & X_N^{\text{HMS}} & J(X_1^{\text{HMS}})
\end{bmatrix}
\]
Harmony Search algorithm (IV)

• **Step 3**: Improvisation a New Harmony from the HM set

  • New harmony vector, \( x' = (x'_1, x'_2, ... , x'_n) \)

  • Three rules:
    - Random selection
    - Memory consideration
      - HMCR: Harmony Memory Considering Rate
      \[
      x'_i \left\{ \begin{array}{ll}
        x_i \in \{x_1^1, x_2^2, ... , x_i^{HMS}\}, & \text{w. p } HMCR \\
        x_i \in X_i, & \text{w. p } 1 - HMCR
      \end{array} \right.
      \]
    - Pitch adjustment
      - PAR: Pitch Adjusting Rate
      \[
      x'_i \left\{ \begin{array}{ll}
        x'_i \pm 1, & \text{w. p } PAR \\
        x'_i, & \text{w. p } 1 - PAR
      \end{array} \right.
      \]
Harmony Search algorithm (V)

- **Step 4**: Updating HM
  - $F(X') < F(X)$?

- **Step 5**: Repeat steps 3 and 4 until the end criterion is satisfied
  - Stop criterion, Number of improvisations (NI)
The m-CPP algorithm (I)

- **Step 1:** Initialization of the optimization problem
  - Employ HS algorithm to find the optimal coverage safe path
  - Minimize $J = J_1 + J_2$
    - Subject to
      - $x_1$ and $x_i$, $i = 1, \ldots, N$
      - $J_1 = K_1 \times \sum_{i=1}^{m} \psi_k^{(i)} + K_2$, $k \in \{135^\circ, 90^\circ, 45^\circ, 0^\circ\}$
      - $J_2 = J_2 \times K_3$, $K_3 > K_1, K_2$, $K_3 \in \mathbb{R}$
        - $J_2' = \mathcal{S}_1 \lor \mathcal{S}_2 \ldots \mathcal{S}_{n-1} \lor \mathcal{S}_n = \bigvee_{i=1}^{n} \mathcal{S}_i$
      - Decision variables
        \[
        X^{(j)} = [x_1, x_2, x_3, \ldots, x_{i-2}, x_{i-1}, x_i],
        \]
        \[
        i = 1, \ldots, N; \quad j = 1, \ldots, \text{HMS}
        \]
        \[
        X^{(1)} = [1, 2, 6, 5, 9, 10, 7, 3, 8, 4]
        \]
        \[
        X^{(2)} = [1, 2, 3, 6, 5, 9, 10, 7, 8, 4]
        \]
The m-CPP algorithm (II)

- **Step 2**: Initialization of the harmony memory (HM)
  - Generate candidate permutations
  - Random Breath Coverage algorithm
  - Numerical example: $X^{(i)} = [1,2,3,6,9,8,7,4,1]$
The m-CPP algorithm (III)

• **Step 3**: Improvisation a New Harmony from the HM set

  • Random selection

  • Memory consideration
    • HMCR: Harmony Memory Considering Rate
      \[ X'_i \left\{ \begin{array}{ll}
      X'_i \in \{ S_i \in X_i \} & \exists s \in X_i, \quad \text{w.p. HMCR} \\
      X'_i \in S_i, & \text{w.p. } 1 - \text{HMCR}
      \end{array} \right. \]
      \[ S = \bigcup_{s \in S} s \]

  • Pitch adjustment
    • PAR: Pitch Adjusting Rate
      \[ X''_i \left\{ \begin{array}{ll}
      X'_i \pm 1, & \text{w.p. PAR} \\
      X'_i, & \text{w.p. } 1 - \text{PAR}
      \end{array} \right. \]
The m-CPP algorithm (IV)

• **Step 4**: Updating HM
  - $J(X') < J(X)$ ?

• **Step 5**: Repeat steps 3 and 4 until the end criterion is satisfied
  - Stop criterion
    - Number of improvisations
    - An admissible number of turns (a hypothesis)
Results achieved (I)

Heuristic approach [7]

m-CPP approach

6.7%
59%
12.5%
Results achieved (II)

- Removing borders [9]
  - Computing time
    - max 2 minutes per area
  - Area coverage
    - Improved
  - Cost
    - Improved for two
    - Worsened for one
Conclusions

- A novel approach to ACPP employing HS algorithm
  - Improved previous approach
  - Improved airspace safety
  - Improved area coverage

- Computation time an issue
  - Large workspaces
  - Divide to conquer
  - Real time computing
Grazie mille!