Exploitation of Rotational Symmetries to Solve the Swarm Initialization Problem

Taryn J. Noone & Norman G. Fitz-Coy

UF UNIVERSITY of FLORIDA
Duke UNIVERSITY
TEXAS The University of Texas at Austin
UC SANTA CRUZ

Overview

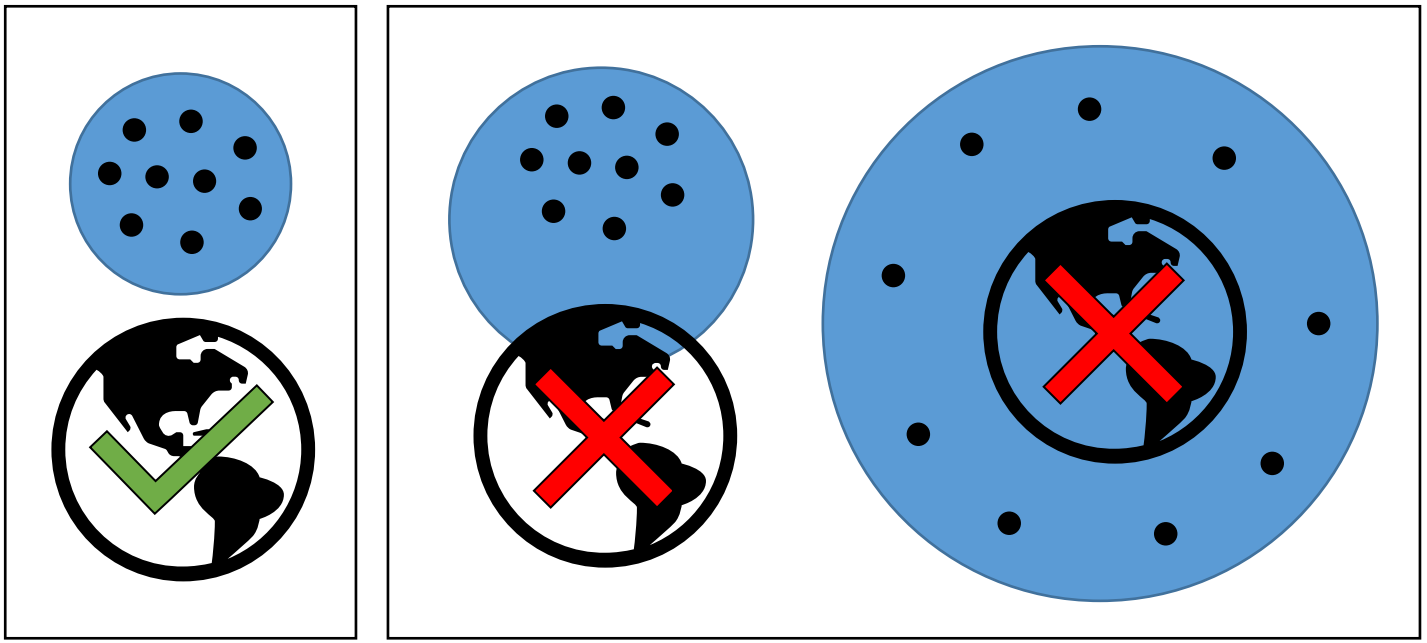
- **We have developed a sound, mathematical basis for solving the swarm initialization problem in a special case.**
  - We will assume a circular swarm trajectory (eccentricity = 0, exact)
  - We define two operations which preserve swarm optimality:
    - Rotation of the swarm within some known space of valid rotations;
    - Transposition of any two satellites at any point in the orbit.
- **Quantization of the solution space** has enabled the use of discretized optimization methods – chiefly, the *Munkres Algorithm*.
- **A staged optimization approach** solves the problem in successively more detailed passes.
- **Computational slowness** is the current major obstacle to implementing this method.
  - Careful algorithm selection;
  - Parallelization of the process;
  - Dividing stages by regularity of use;

Principal Definitions

In prior discussions, a swarm was defined to be a close-flying formation of satellites for which the following quantities could be defined:

- **Swarm envelope**
  - A **closed, convex set of points with no holes, gaps, or voids**, which contains all satellites in the swarm.
    - Mathematically speaking**, the boundary of the envelope must be simply connected.**
  - The swarm envelope **may not contain points inaccessible to the satellites** (e.g., points at or below Earth's surface).
  - For this discussion, we will use a **spherical envelope**.
  - Must include a **reference point** that marks the envelope center.
    - This point **need not to be an element** of the swarm envelope.
    - For this discussion, we will use the **center of the sphere**.

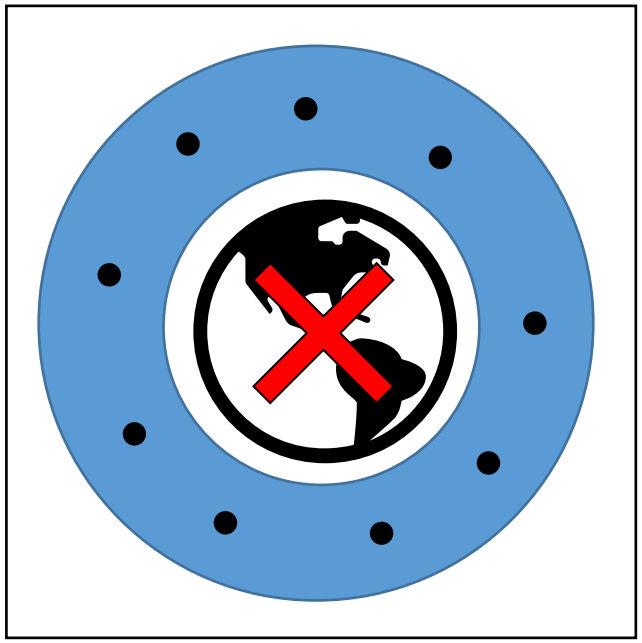
Principal Definitions



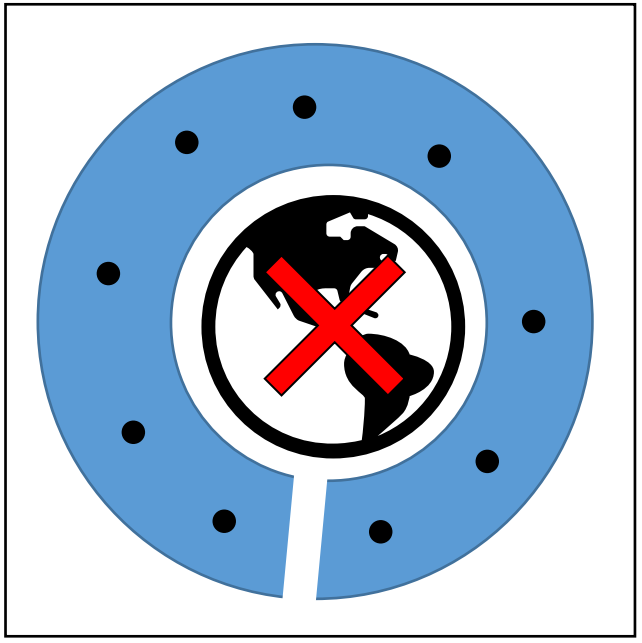
VALID

CONTAIN INACCESSIBLE POINTS

Principal Definitions



CONTAINS A VOID

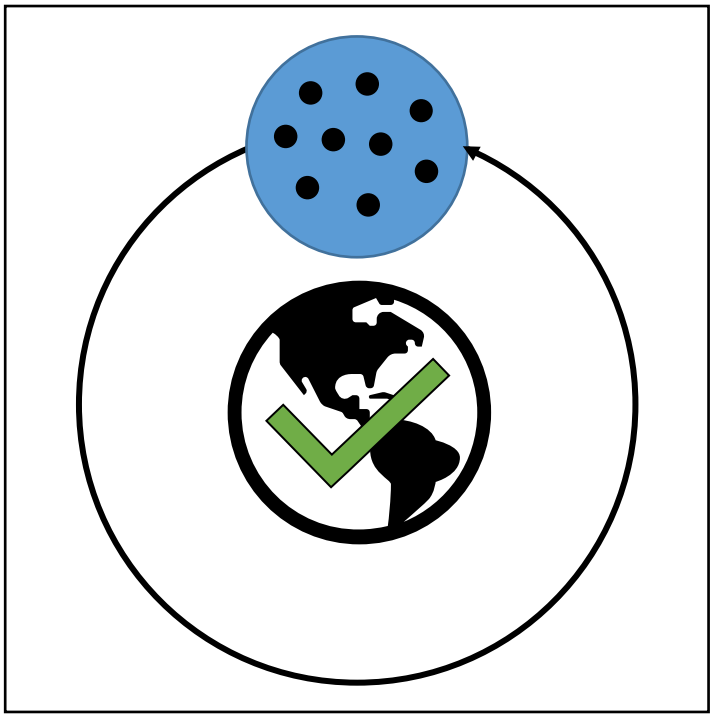


NOT CONVEX

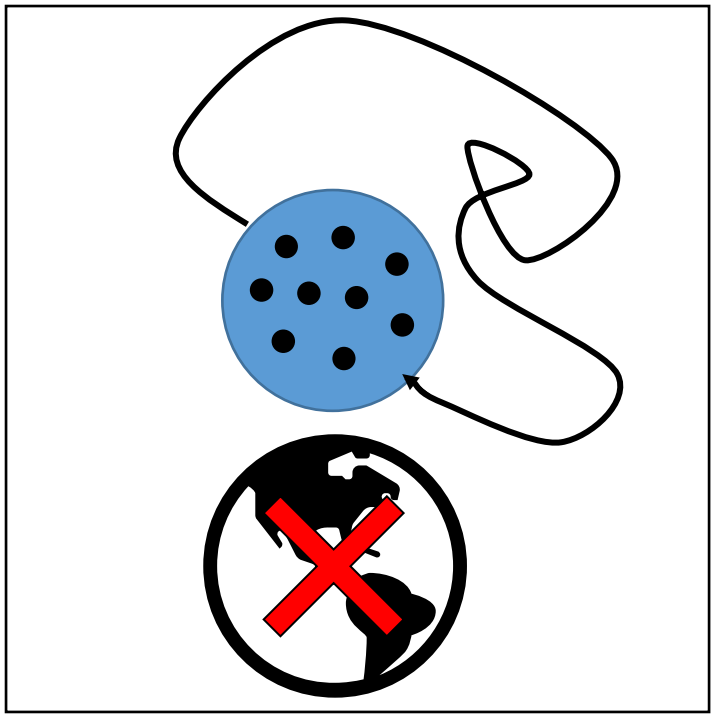
Principal Definitions

- **Swarm trajectory**
  - A **user-defined function of time** which specifies the position of the swarm envelope relative to any point with known coordinates.
    - This latter point can be taken to be the **center of the Earth**.
    - Position measured to the **envelope-relative reference point**.
  - If necessary, this definition may include a function to specify **the orientation of the swarm envelope**.
    - Unnecessary in the case of a spherical envelope.
  - While it is not a strict requirement in the general case, **trajectories should follow spacetime geodesics**.
  - For this discussion, we will assume that the swarm trajectory **can be approximated by a Keplerian orbit of zero eccentricity**.

Principal Definitions



FOLLOWS A
GEODESIC PATH



FOLLOWS ONLY
ITS DREAMS

Principal Definitions

- **Swarm distribution**
  - The **arrangement of satellites** within the sphere envelope as defined relative to the envelope-relative reference point.
  - The most abstract component of the satellite swarm, incorporating mission-specific parameters, an unknown number of degrees of freedom, and the swarm cost functional.
  - For this discussion, we will assume that the swarm distribution **can be rotated** about the envelope relative reference point.
- **Swarm centroid**
  - A function of the satellite positions which **determines the true center of the swarm**.
  - Distinct from the envelope-relative reference point, as the centroid is **determined by satellite positions, not envelope geometry**.
  - For this discussion, we will use the **arithmetic mean** of all satellite positions to define the swarm centroid.

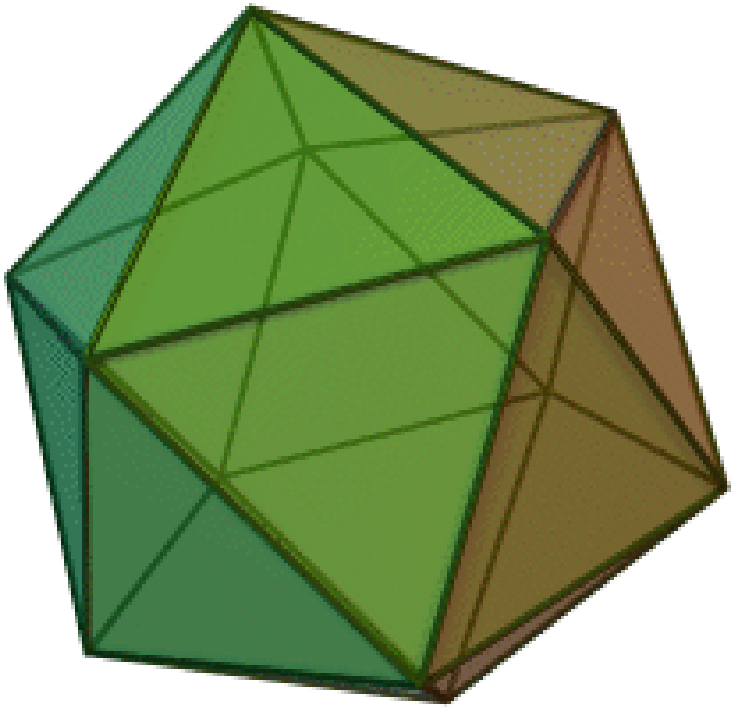
Test Swarm Geometry

For this discussion, we will default to our test case of **twelve satellites**:

- The swarm envelope is a **sphere of radius ρ** with reference at its center.
- The swarm trajectory is a **circular orbit of radius a**.
- The satellites in the swarm should be distributed at the **vertices of a regular icosahedron**.
  - The swarm may be **transformed by any rotation** about its center.
- The swarm centroid is defined using the arithmetic mean:

$$\vec{r}_O(t) = \frac{1}{12}\sum_{i=1}^{12} \vec{r}_i(t)$$

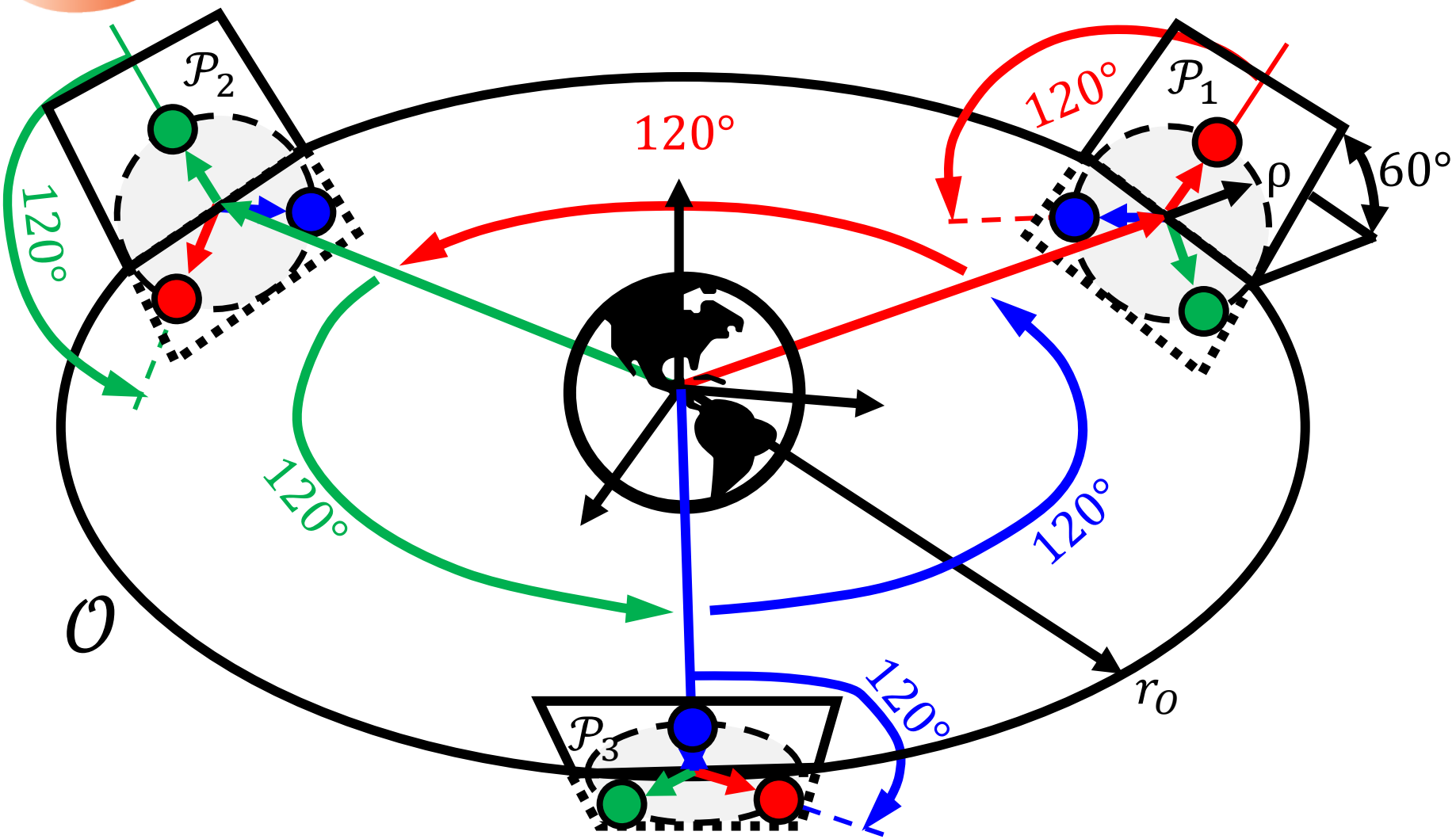
Test Swarm Geometry



A rotating icosahedron is still an icosahedron.

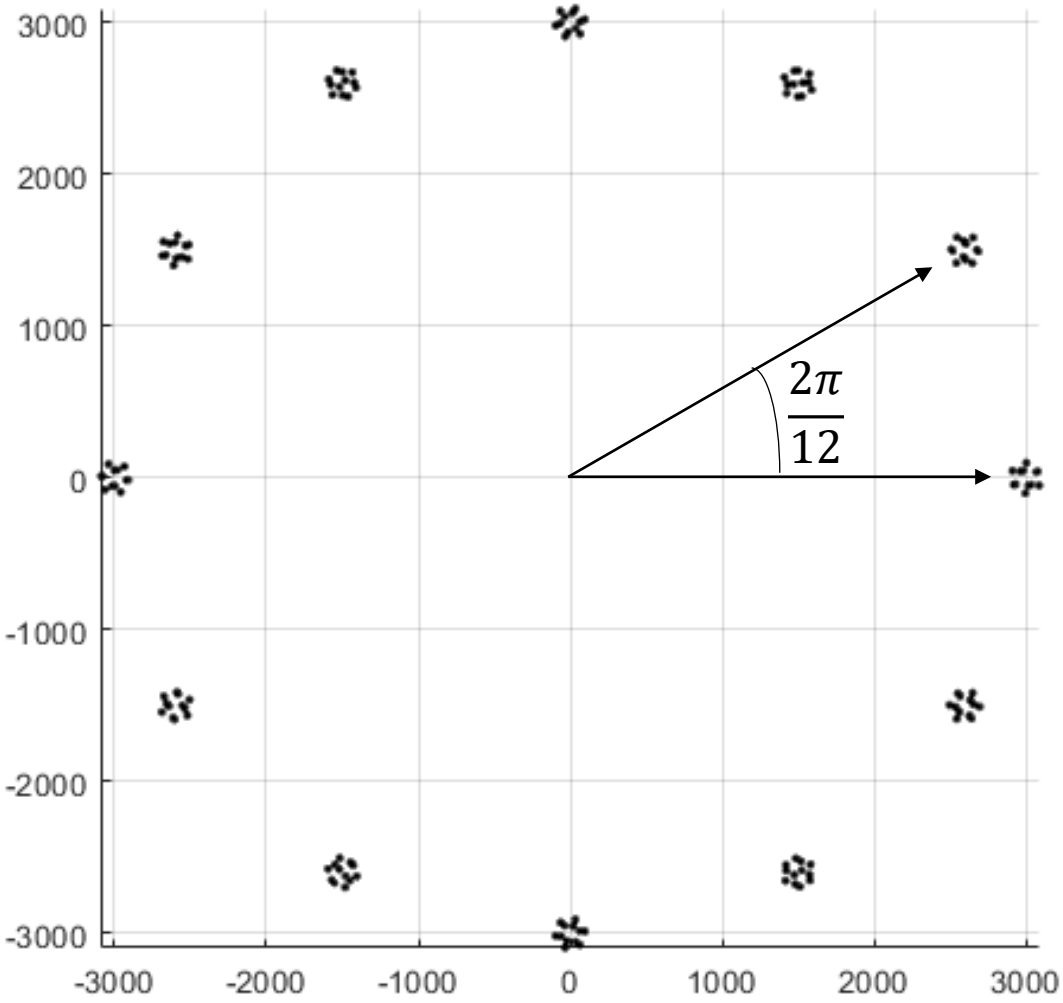
Animation from Wikipedia.

Test Swarm Configuration

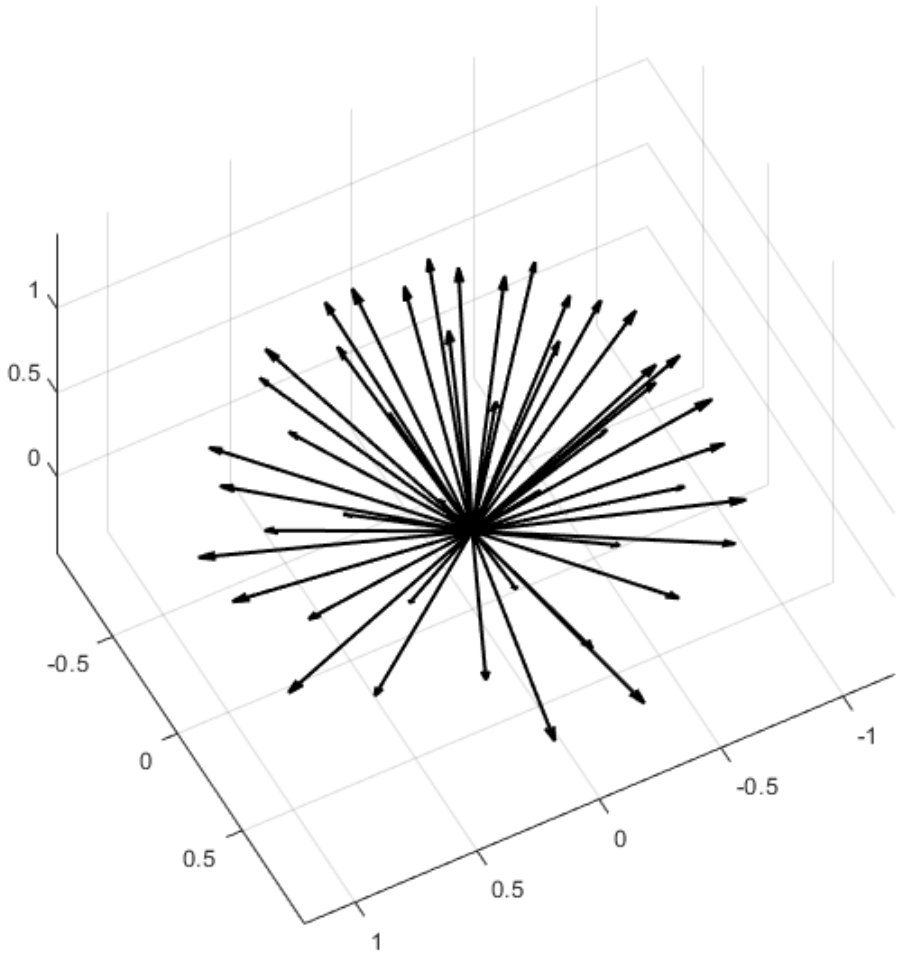


UF UNIVERSITY of FLORIDA · Duke UNIVERSITY · TEXAS The University of Texas at Austin · UC SANTA CRUZ

Formation Chains



Formation Chains



Formation Chains

- Note that n successive rotations by $2\pi/n$ radians about a given axis will result in a net angle of 2π radians.
- We may, however, rotate by *multiples* of $2\pi/n$ radians.
  - Recognizing that a rotation by 2π radians is equivalent to no rotation, it follows that rotations by multiples of $n+1$ to $2n-1$ are equivalent to rotations by **multiples of 1 to $n-1$.**
  - We may also consider the negative multiples (i.e., $-(n-1)$ to -1).
  - Rotations by a multiple of 0 need only be considered about a single axis, since this corresponds to zero rotation.
- With two distinct rotations to check (the starting attitude and the change in attitude), we must check $\mathbf{4m^2n(n-1)+1}$ **formation chains**.
- For 46 axes and 12 satellites, this corresponds to 1,117,249 chains.
- Some of these chains may be sufficiently close to others to qualify as "duplicate." **We may eliminate these duplicate chains**.
  - This process takes several days to a week or more on a single CPU, but only needs to be done once (the results may be saved and re-used).

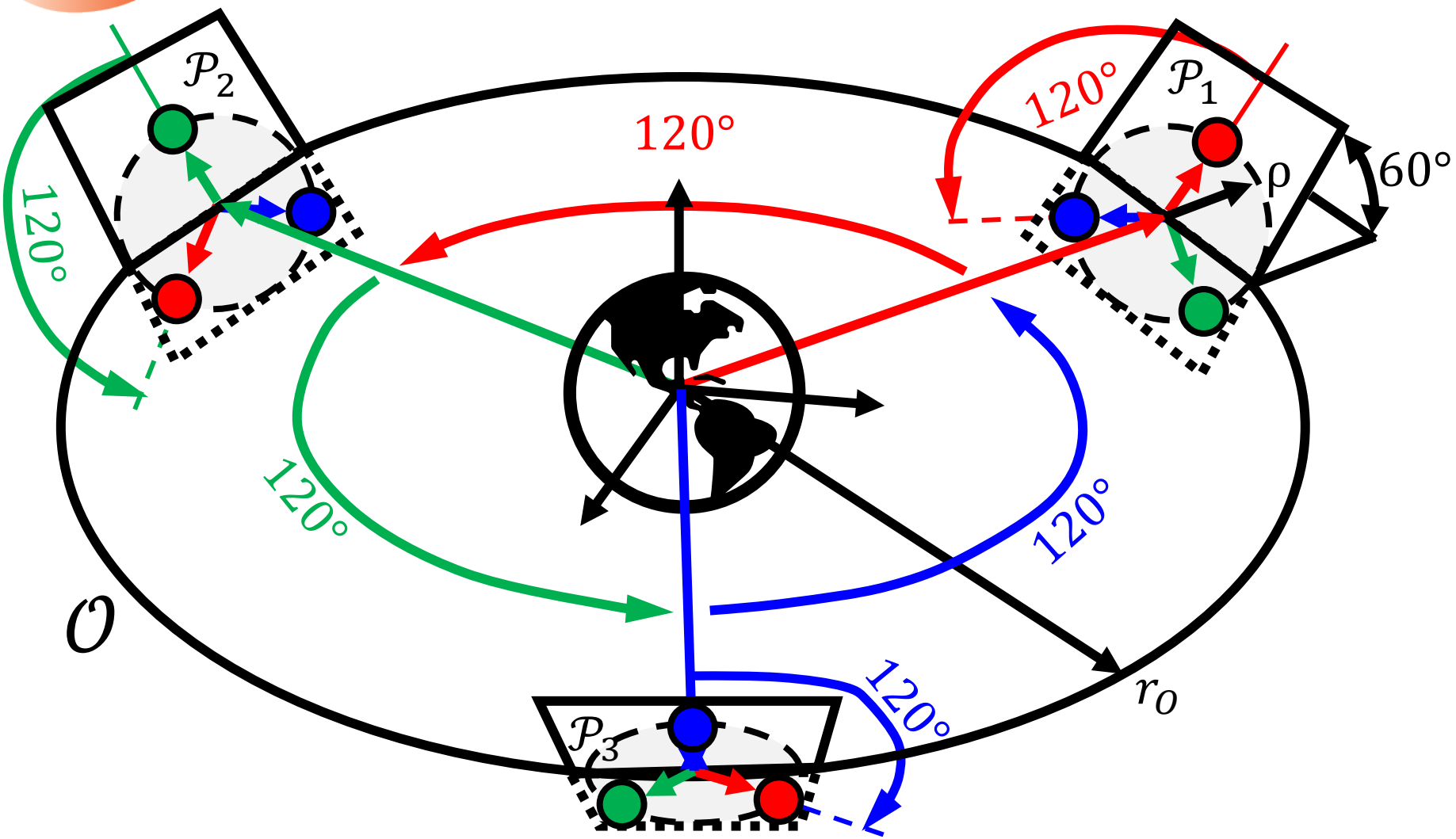
Formation Chains

- For 46 axes and 12 satellites, **with duplicate elimination**, we obtain **between 5,000 and 10,000** formation chains depending on tolerance.
- **These formation chains are saved** and used in the next step.
  - Note that, because these chains depend only on the selected value of the rotation, this process is **invariant with respect to the radii of the formation, or its orbit.**

Selecting Orbits

- Once a group of formation chains are selected, we must then pull orbits from them by **selecting the first two formations out of the chain**.
- For this pair, we may connect **each satellite from the first formation to any position from the second.**
  - Connecting the ith satellite to the jth position yields an orbit. Comparing this orbit to the formation chain yields a contribution to a cost function, J_{ij}.
  - For each i and j, we select the two formations with minimal J_{ij}.
- Note that, for each pair, we may propagate the assignment to another formation, so we must compare the values of J_{ij} for each formation in the chain.
- The optimal assignment may be determined using the Munkres Algorithm

Selecting Orbits



Selecting Orbits

- The Munkres algorithm runs in $O(n^4)$ time, so we are currently investigating ways to avoid running the Munkres algorithm in cases where it would clearly produce a suboptimal result.
  - For example, if the sum of the n smallest elements in the cost matrix, J is greater than the current best result, we may skip the algorithm.
- Once the best result has been found, that formation chain is selected, along with the assignment function determined by Munkres and the orbits it generates.
- Those orbits are passed on to the next stage of optimization, which uses continuous optimization to further refine the initial velocities and minimize the swarm cost functional.

Next Steps

- Implement code to run the orbit selection protocol.
- Implement code to refining stage.
- Run this with known test cases, such as the LISA configuration.
- Compare performance for low-n vs. high-n cases.