Correctness Analysis and Optimality Bounds of Multi-spacecraft Formation Initialization Algorithms

Mike Schuresko and Jorge Cortés

***Abstract*— This paper considers formation initialization for a class of autonomous spacecraft operating in deep space with arbitrary initial positions and velocities. Formation initialization is the task of getting a group of autonomous agents to obtain the relative and/or global dynamic state information necessary to begin formation control. We associate a "worst-case total angle traversed" optimality notion to the execution of any formation initialization algorithm, and present performance bounds valid for any correct algorithm. We design the SPATIAL SPACECRAFT LOCALIZATION ALGORITHM and the WAIT AND CHECK ALGORITHM, analyze their correctness properties and characterize their performance in terms of worst-case optimality and execution time.**

I. INTRODUCTION

Motivation and problem statement: Deploying large structures in space requires multiple spacecraft to coordinate their activities, due, in part, to the limited payload capabilities of launch vehicles. One application that requires such coordination is the deployment of large-baseline interferometers for science imaging missions. Key aspects of spacecraft coordination which are likely to be used in a broad variety of contexts include: (i) formation initialization, i.e., the establishment and maintenance of relative dynamic state information (e.g. relative positions and velocities) and/or on-board inter-spacecraft communication; (ii) formation acquisition, i.e., making the group of spacecraft attain a desired geometry; and (iii) formation regulation and tracking, i.e., maintaining fixed inter-spacecraft range, bearing, and inertial attitudes with high accuracy along the execution of a desired trajectory.

In this paper, we focus our attention on the formation initialization problem. This problem is especially important for spacecraft operating in deep space, where conventional Earth-based GPS does not provide sufficiently accurate position information. Here, we consider a spacecraft model motivated, in part, by the design possibilities of NASA's "Terrestrial Planet Finder" mission. Our spacecraft model is similar to the one proposed in [1]. The spacecraft have laser-based directional relative position sensors, like the kind described in [2], which require two sensors to lock on to each other before getting a position measurement. Each of the spacecraft has a sun-shield which must be oriented so as to protect sensitive astronomical instruments from solar radiation. The spacecraft are assumed to be in deep space, far from the effects of gravitational curvature.

Literature review: A fairly extensive bibliography of missions which plan to use spacecraft formation flying can be found in [3]. These include Terrestrial Planet Finder [4], EO-1 [5], TechSat-21 [6] and Orion-Emerald [7]. A driving motivation behind formation flying research is that of large aperture adaptive optics in space, e.g. [4]. Optical devices such as the ones described in [8] could combine the advantages of multi-mirror adaptive optics with those of space telescopes. A good overview on current research on formation flying for optical missions is contained in [2].

The majority of the work on control algorithm design has focused on formation acquisition and tracking. A survey of algorithms is given in [9]. Leader-following approaches, e.g. [10], [11], and virtual structures approaches, e.g. [12], have been used to prescribe overall group behavior by specifying the behavior of a single leading agent, either real or virtual. Motion planning and optimal control problems are analyzed in [13]. The only work known to us that has dealt in detail with formation initialization is [1].

Statement of contributions: The contributions of this paper are twofold. On the one hand, we provide optimality bounds on the performance of any correct formation initialization algorithm. Our analysis consists of a systematic study of optimality of algorithms, both in two and three dimensions, with regard to worst-case total angle rotated by any member of the group of spacecraft. As a byproduct of our analysis, we provide justification for the **Opposing Sensor Constraint** in [1] by showing that optimal algorithms exist which invoke it. Our optimality bounds give rise to necessary conditions, which we use to show that the rotation phases of the algorithm presented in [1] fail to achieve formation initialization.

On the other hand, we present two original formation initialization algorithms. The SPATIAL SPACECRAFT LOCALIZATION ALGORITHM achieves formation initialization through a simple sequence of rotational maneuvers, each of which sweeps a region of a particular partition of space. The WAIT AND CHECK ALGORITHM performs a sequence of rotational maneuvers interspersed with carefully chosen pauses in order to achieve a nearly-optimal formation initialization solution. For both algorithms, we assess their correctness, and formally characterize their performance with regards to the optimality measures mentioned above. It should be noted that, from a practical viewpoint, the pauses employed by the WAIT AND CHECK ALGORITHM make the SPATIAL SPACECRAFT LOCALIZATION ALGORITHM more amenable to actual implementations.

Organization: Section II presents a set of definitions which will be used throughout the remainder of the paper. In Section III we give necessary conditions for the correctness of any candidate solution to this problem, from which we derive lower bounds for the optimality of any working

M. Schuresko and J. Cortés are with the Department of Applied Mathematics and Statistics, Baskin School of Engineering, University of California at Santa Cruz, CA 95064, USA {mds,jcortes}@soe.ucsc.edu

solution. These conditions will be used to analyze an algorithm from the literature. Section IV presents three provable formation initialization algorithms, including an algorithm for formation initialization in 2 dimensions in Section IV-A, a simple algorithm for 3 dimensions in Section IV-B and an algorithm that gets close to our optimality bounds at the expense of long wait times in Section IV-C.

II. Preliminaries

Each spacecraft consists of a rigid body containing instruments on one side, which need to be shielded from the sun (see Fig. 1). To serve this purpose, a *sun shield*

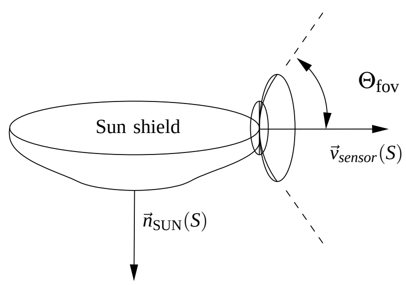

Fig. 1. Configuration of spacecraft geometry, and body frame definition.

is mounted to the spacecraft body on the side opposite the instruments. The *sun shield normal vector*, $\vec{n}_{\text{SUN}}(S)$, indicates the direction of the sun shield of spacecraft S. We make the approximation that the sun is an infinite distance away, and therefore the vector to the sun, $\vec{v}_{\text{SUN}}$, is the same for each spacecraft. In order to operate without damaging the instrumentation, each spacecraft must maintain the constraint $\vec{n}_{\text{SUN}}(S)\cdot\vec{v}_{\text{SUN}} \geq \cos(\Theta_{\text{sun}})$ for some pre-specified angle Θ_{sun} at all times. Relative position and velocity measurements between two spacecraft are made through the *metrology sensors* of the two craft. The metrology sensor of spacecraft S senses within a conical region (C_S) with a half angle of Θ_{fov} (assumed here to be greater then $\frac{\pi}{4}$ unless otherwise stated). The *sensor cone centerline* of S is an infinite ray down the axis of rotational symmetry of the sensor cone defined by the unit vector $\vec{v}_{\text{SENSOR}}(S)$. Accurate orientation information is available for all spacecraft through measurements of what are known as reference stars. Thus we only have to worry about obtaining relative position and velocity information for each spacecraft. The spacecraft are placed such that the curvature of earth's gravitational field has a negligible effect (for instance a Lagrange point). We therefore assume that if no spacecraft undergo translational acceleration then the spacecraft move with constant (initially unknown) velocity in straight lines relative to each other.

Definition 2.1: The *global frame of reference* is an arbitrary orthonormal frame, $GF = \{X_g, Y_g, Z_g\}$, where $X_g = \vec{v}_{\text{SUN}}$. For a spacecraft S, let P_S be the position of the center of mass of S in the frame GF. The *center of mass frame* of S (denoted $CMF(S)$) corresponds to translating the global frame GF to P_S.

Definition 2.2: Let S be a spacecraft. The body frame, $BF(S) = \{\hat{X}_S, \hat{Y}_S, \hat{Z}_S\}$ is defined by $\hat{X}_S = \vec{n}_{\text{SUN}}(S)$, $\hat{Z}_S = \vec{v}_{\text{SENSOR}}(S)$ and $\hat{Y}_S = \hat{Z}_S \times \hat{X}_S$. In this frame, $\{0,0,0\}$ is at the center of mass, $CM(S)$, of the spacecraft (see Fig. 1).

Now that we have these reference frames, we can define the sensor cone $C_S : \mathbb{R}^3 \times SO(3) \to 2^{\mathbb{R}^3}$ of spacecraft S as

$$C_S(P_S, M_S) = \{\vec{x} \in \mathbb{R}^3 \;:\; \frac{[0,0,1]^T M_S(\vec{x} - P_S)}{\|\vec{x} - P_S\|} \leq \cos(\Theta_{\text{fov}})\}. \tag{1}$$

When it is clear from the context, we will use the simpler notation C_S. In order to get a relative position reading between two spacecraft, S_1 and S_2, S_1's metrology sensor must point at S_2. This condition is called *sensor lock*. Formally, two spacecraft, S_1 and S_2, achieve **sensor lock** if and only if $P_{S_1} \in C_{S_2}$ and $P_{S_2} \in C_{S_1}$.

The algorithms we present all require the n spacecraft performing the algorithm to be split into two groups, G_1 and G_2 such that $G_1 \cup G_2 = S_1, ..., S_n$ and $G_1 \cap G_2 = \emptyset$. This can either be done a priori before launch or (preferably) with a distributed algorithm prior to running formation initialization (see [14]).

The state of each spacecraft $S \in \{S_1, \cdots, S_n\}$ can be described by $(P_S, M_S) \in \mathbb{R}^3 \times SO(3)$. The dimensionality of the network of spacecraft ($\{S_1, \cdots, S_n\}$) is therefore $6n$. M_S transforms $BF(S)$ onto $CMF(S)$ and P_S defines the translation between $CMF(S)$ and GF. The spacecraft are fully actuated.

Two spacecraft, S_1 and S_2, are said to be maintaining the **Opposing Sensor Constraint** if $\vec{v}_{\text{SENSOR}}(S_1) = -\vec{v}_{\text{SENSOR}}(S_2)$.While this constraint is not strictly necessary for a correct solution to the Formation Initialization problem, we will show, in Section III, that it is a convenient and desirable constraint to work with. Note that this does not fully constrain the relative orientation of S_1 with respect to S_2. When specifying an algorithm requiring the **Opposing Sensor Constraint**, we will often specify the more restrictive constraint

$$M_{S_1}^{-1} M_{S_2} = M_{\text{opp}} = \text{diag}(1, -1, -1).$$

We call this constraint the **Opposing Frame Constraint**.

Lemma 2.3: In addition to maintaining the **Opposing Sensor Constraint**, the **Opposing Frame Constraint** also guarantees that if spacecraft S_1 verifies the sun-angle constraint, then S_2 also verifies it.

A. Algorithm definition

In this section we formally define what we mean by an "algorithm." In the next definitions, let D_{msg} be the set of possible messages a spacecraft can communicate at any instant, and let $D_{\text{sensor}} = (\mathbf{Z}_2 \times \mathbb{R}^3 \times \mathbb{R}^3)^n$ be the set of possible sensor cone readings for a spacecraft.

Definition 2.4 (Algorithm notion): An algorithm is a tuple $A_S = (\text{STATE}_{S,0}, F_S, G_S, \delta t_{step})$, where $\text{STATE}_{S,0} \in D_{\text{STATE}}$, the initial internal state of spacecraft S, contains no information about the location of the other spacecraft and F_S is a map of the form

$$\begin{aligned} F_S : \mathbb{R} \times SO(3) \times D_{\text{STATE}} &\to \mathbb{R}^3 \\ (t, M_S, \text{STATE}_S) &\mapsto \omega_S \end{aligned}$$

and G_S is a map of the form

$$G_S : \mathbb{R} \times SO(3) \times D_{\text{STATE}} \times D_{\text{msg}}^{n-1} \times D_{\text{sensor}} \to D_{\text{STATE}} \times D_{\text{msg}}$$
$$(t, M_S, \text{STATE}_S, \text{MSG}_{S,\text{in}}, \text{SENSOR}_S) \mapsto (\text{STATE}_S, \text{MSG}_{S,\text{out}}).$$

Definition 2.5 (Execution of an algorithm): An execution by a spacecraft S of an algorithm $A_S = (\text{STATE}_S, F_S, G_S, \delta t_{step})$ during the time interval $[t_0, t_f]$ is the pair of trajectories $t \in [t_0, t_f] \to (P_S(t), M_S(t)) \in \mathbb{R}^3 \times SO(3)$ and $\text{STATE}_S : [t_0, t_f] \to D_{\text{STATE}}$ defined as follows:

- $\dot{P}_S(t) = V_S$, for some constant $V_S \in \mathbb{R}^3$;
- $\dot{M}_S(t) = F_S(t, \widehat{\text{STATE}_S(t)}) M_S(t)$, $t \in [t_0, t_f]$, where $\hat{\omega}$ is the matrix operator for the cross product with $\omega \in \mathbb{R}^3$;
- STATE_S is the piecewise constant function defined by

$$\text{STATE}_S(t_{i+1}) =$$
$$G_S((t_i, M_S(t_i), \text{STATE}_S(t_i), \text{MSG}_{S,\text{in}}(t_i), \text{SENSOR}_S)(t_i))$$

  for $i = 0, \ldots, m-1$, with $t_0, t_1, \ldots, t_m \in [t_0, t_f]$ a finite increasing sequence, where $t_i = k\,\delta t_{step}$ for some $k \in \mathbb{N}$ or t_i corresponds to the time instant when a change occurs in the value of the sensor cone readings. The initial value of $\text{STATE}_S(t_0)$ is $\text{STATE}_{S,0}$.

The lack of concrete specification of D_{msg} and D_{STATE} reflects our intent to provide lower bounds on algorithmic performance for spacecraft with a wide range of computational and communication capabilities. In practice, the working algorithms we present in Section IV require basic computational capabilities on the part of each spacecraft.

B. Total angle traversed and solid angle covered

In this section we present definitions related to our notion of an optimal solution to the formation initialization problem.

1) Definition of total angle traversed during an algorithm: In 3 dimensions, recall that $M_S = [m_x, m_y, m_z]$ is an orthonormal basis matrix representing the orientation of spacecraft S. From Equation 8.6.5 of [15] we have the formula for $\hat{\omega} = \dot{M}_S M_S^{-1}$.

The total angle traversed during the execution of an algorithm in 3 dimensions is therefore

$$\int_{t=t_0}^{t_f} \sqrt{\hat{\omega}_{1,2}^2 + \hat{\omega}_{1,3}^2 + \hat{\omega}_{2,3}^2}\, dt.$$

One can think of the 2-D problem as the 3-D problem with rotations confined to the $\{Y, Z\}$ plane. Under this constraint, the previous expression reduces to

$$\int_{t=t_0}^{t_f} |\hat{\omega}_{2,3}|\, dt.$$

2) Definition of solid angle traversed during an algorithm: Sometimes it is useful to discuss the total solid angle covered by the sensor cone (C_S) of a spacecraft S performing a formation initialization algorithm in 3 dimensions.

If a spacecraft, S, with sensor cone field of view Θ_{fov} rotates by an angle of π about an axis initially at an angle of $\Theta > \Theta_{\text{fov}}$ with respect to $\vec{v}_{\text{SENSOR}}(S)$, the new solid angle covered in this sweep (i.e. the solid angle covered during some portion of the sweep that was not in C_S at the time the sweep started) can be found by tracing a band about the unit sphere and calculating its area. See Figure 2 for clarification.

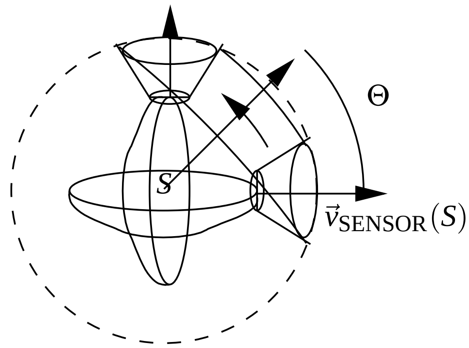

Fig. 2. Method to compute rate of change of solid angle swept.

Recall that the solid angle of a cap of half angle α is $\int_0^\alpha 2\pi \sin(t)\, dt$. The area of this band can be found by subtracting caps of half angles $\Theta - \Theta_{\text{fov}}$ and $\pi - \Theta - \Theta_{\text{fov}}$ from the unit sphere and dividing by 2. Dividing by π gives a rate of change of coverage of solid angle for this operation.

Definition 2.6: Define the function $f_{\text{solid}}(\omega)$ to be $f_{\text{solid}}(\omega) = 2\|\omega \times \vec{v}_{\text{SENSOR}}(S) \sin(\Theta_{\text{fov}})\|$ when $\arccos(\frac{\omega \cdot \vec{v}_{\text{SENSOR}}(S)}{\|\omega\|}) > \Theta_{\text{fov}}$ and $f_{\text{solid}}(\omega) = \|\omega \times \vec{v}_{\text{SENSOR}}(S) \sin(\Theta_{\text{fov}})\| + \|\omega\| - |\omega \cdot \vec{v}_{\text{SENSOR}}(S)|$ for all other ω.

The total solid angle covered by a spacecraft, S, performing an algorithm, A, between times t_0 and t is

$$F_{\text{solid}}(t) := \int_{t=t_0}^{t} f_{\text{solid}}(\omega)\, dt.$$

We will consider the total solid angle covered by S during the course of the algorithm to be $F_{\text{solid}}(t_f) + \alpha_0$ where t_f is the earliest time at which formation initialization is guaranteed to be complete and $\alpha_0 = 2\pi(1 - \cos(\Theta_{\text{fov}})$ is the solid angle contained in $C_S(t_0)$ at time t_0.

Remark 2.7: Note that $0 \leq f_{\text{solid}}(\omega) \leq 2\|\omega\| \sin(\Theta_{\text{fov}})$. •

Analogously, the total angle covered by a spacecraft S performing an algorithm A in 2d between times t_0 and t is

$$F_{\text{angle}}(t) := \int_{t=t_0}^{t} |\omega|\, dt.$$

C. Formation initialization problem

Formation initialization solutions entail establishing communication and/or relative position information. Frequently they also involve moving the spacecraft to an initial formation from which another formation control algorithm can take over. Here we restrict ourselves to the establishment of relative position and velocity information between each pair of spacecraft. We assume that this information can come from any combination of direct sensor readings, odometry and communication with other spacecraft.

Definition 2.8: Let $[t_s, t_f]$ be the duration of time during which a formation initialization algorithm runs. Let $G(t)$ be the **relative position connectivity network** at time t, defined by $G(T) = (V, E)$ where $v(S_i) \in V$ correspond to the spacecraft S_i, and the edge $(v(S_i), v(S_j))$ is in E if and only if spacecraft S_i and S_j are in a state of sensor lock. A solution to the formation initialization problem is one that guarantees that the graph $\cup_{t\in[t_s,t_f]} G(t)$ is connected, so long as no two spacecraft collide by t_f.

The multi-spacecraft algorithm proposed in [1] to solve formation initialization is briefly described in Table I. We discuss its correctness in Section III-A.

Name:	Formation Initialization Algorithm
Goal:	Solve the formation initialization problem assuming using translation and rotation
Assumes:	Assumptions in Section II.

1: **if** $S_i \in G_1$ **then**
2: Rotate to align M_{S_i} with I_3
3: **else**
4: Rotate to align M_{S_i} with M_{opp}
5: **end if**
6: Wait for common start time t_s
7: Rotate by 3π about X_{S_i}.
8: Rotate $-\Theta_{\text{tilt}}$ (in this case 25 degrees) about Y_{S_i}.
9: Rotate $2\Theta_{\text{tilt}}$ about Y_{S_i}.
10: Rotate by π about X_{S_i}.
11: Rotate $2\Theta_{\text{tilt}}$ about Y_{S_i}.
{This is the end of the rotational component of the algorithm}
12: Rotate $-\Theta_{\text{tilt}}$ about Y_{S_i}.
13: Wait for some time $t_{\text{near field}} > 0$
14: **if** $S_i \in G_1$ **then**
15: Begin translating along Z_{S_i} with speed v_{max}, where v_{max} is the maximum relative velocity between any two craft.
16: **end if**

TABLE I

FORMATION INITIALIZATION ALGORITHM PROPOSED IN [1]. WHILE $t_{\text{NEAR FIELD}}$ IN STEP 13 IS CAREFULLY SPECIFIED IN [1], THE ACTUAL VALUE OF $t_{\text{NEAR FIELD}}$ IS NOT RELEVANT TO OUR ANALYSIS.

III. CORRECTNESS AND OPTIMALITY OF FORMATION INITIALIZATION ALGORITHMS

In Section III-A, we provide a necessary condition for the correctness of any formation initialization algorithm. Then, in Section III-B, we proceed to use this condition as the basis for a series of optimality bounds. We also present optimality results which justify the **Opposing Sensor Constraint** and allow us to more easily reason about the n spacecraft case (where $n > 2$).

A. Necessary conditions for correctness

Theorem 3.1 presents a necessary condition for the correctness of a formation initialization algorithm. Theorem 3.3 demonstrates the utility of this result by using it to analyze an existing algorithm from the literature.

Theorem 3.1: Let S be executing a correct formation initialization algorithm in d dimensions, with $d \in \{2,3\}$. For every $v \in \mathbb{R}^d$, let t_v be the first time such that $v \in C_S(t_v) = C_S(P_S(t_v), M_S(t_v))$. Then, there must exist $t^* > t_v$ such that $-v \in C_S(t^*)$.

Proof: For simplicity, let $\text{vers}(u) = u/\|u\|$, for $u \in \mathbb{R}^d$. Consider two spacecraft, S_1 and S_2. S_2 travels in the plane defined by it's velocity (V_{S_2}), and $p_{\text{closest}}(S_1,S_2)$, where $p_{\text{closest}}(S_1,S_2)$ is the point of closest approach between S_1 and S_2 in $CMF(S_1)$. At time t S_2 makes an angle with $p_{\text{closest}}(S_1,S_2)$ of $\arctan(\frac{\|V_{S_2}\|}{\|p_{\text{closest}}(S_1,S_2)\|} t + t_0)$ for some t_0. S_2's initial conditions can be chosen to match any arbitrary V_{S_2}, $p_{\text{closest}}(S_1,S_2)$ and t_0. Because of this, given an ε and times, t_1 and t_2, $\text{vers}(P_{S_2})$ can be made to stay within an angle of ε of $-\text{vers}(V_{S_2})$ until time t_1, and move to within an angle of ε of $\text{vers}(V_{S_2})$ by t_2. Let t_1 be the first time at which the minimum angle between any ray in $C_{S_1}(t_1)$ and $\text{vers}(-V_{S_2})$ is less then or equal to ε and t_2 be the first time at which $C_{S_1}(t_2)$ includes $\text{vers}(-V_{S_2})$. In order to ensure S_1 finds S_2, $C_{S_1}(t^*)$ must include $\text{vers}(V_{S_2})$ at some time $t^* > t_1$. Since ε was picked arbitrarily and the sensor cone is always closed, $C_{S_1}(t^*)$ must include $\text{vers}(V_{S_2})$ at some time $t^* > t_2$. ■

Using this result, we analyze the correctness of the formation initialization algorithm proposed in [1] and summarized in Table I. Let $\{e_1, e_2, e_3\}$ be the canonical basis for $\mathbb{R}^3$. Given $S \in G_1$, let

$$R_{down}(S) = \{v \in CMF(S) \,:\, \frac{v \cdot e_1}{\|v\|} > \cos(\frac{\pi}{2} - \Theta_{\text{fov}})\} \cap$$
$$\{v \in CMF(S) \,:\, \frac{v \cdot (e_1 \sin(\Theta_{\text{tilt}}) + e_3 \cos(\Theta_{\text{tilt}}))}{\|v\|} < \cos(\Theta_{\text{fov}})\}.$$

In other words, with regards to Table I, R_{down} is the set of points in the sensor cone at the end of Step 8 that were not in the sensor cone at any point during Step 7, nor at the end of Step 11.

Lemma 3.2: With the conventions of Table I, $R_{down}(S)$ is non-empty so long as $\Theta_{\text{tilt}} + \Theta_{\text{fov}} < \pi - \Theta_{\text{fov}}$.

Proof: Let $\varepsilon > 0$ such that $\pi - \Theta_{\text{fov}} - \varepsilon > \Theta_{\text{tilt}} + \Theta_{\text{fov}}$. Let $\theta = \Theta_{\text{fov}} + \varepsilon$, and define $v := -e_3 \cos(\theta) + e_1 \sin(\theta)$. We show next that $v \in R_{down}(S)$. This is because

$$\frac{v \cdot e_1}{\|v\|} = \cos(\frac{\pi}{2} - \Theta_{\text{fov}} - \varepsilon) > \cos(\frac{\pi}{2} - \Theta_{\text{fov}}),$$
$$\frac{v \cdot (e_1 \sin(\Theta_{\text{tilt}}) + e_3 \cos(\Theta_{\text{tilt}}))}{\|v\|} = -\cos(\theta + \Theta_{\text{tilt}})$$
$$= \cos(\pi - \theta - \Theta_{\text{tilt}}) < \cos((\Theta_{\text{tilt}} + \Theta_{\text{fov}}) - \Theta_{\text{tilt}}) \leq \cos(\Theta_{\text{fov}}).$$

■

Theorem 3.3: The algorithm stages described in Steps 1-12 of Table I are not, by themselves, sufficient to solve the formation initialization problem.

Proof: Let $S \in G_1$ perform this algorithm. By Theorem 3.1, for any vector v, $C_S(t)$ must contain $-v$ at least once before the last time $C_S(t)$ contains v. But each $v \in R_{down}$ is last in $C_S(t)$ during Step 9, and no $v \in \{u \in CMF(S) \,:\, -u \in R_{down}\}$ is in $C_S(t)$ before Step 10. Thus $R_{down}(S)$ does not satisfy this condition. ■

Obviously, the correctness of the algorithm in Table I hinges on v_{max}. In fact, if v_{max} is known, Steps 13-16 by themselves provide a correct formation initialization algorithm. It should be noted, however, that these steps were designed to handle an effect we do not consider in this paper (cf. Definition 2.2 and equation (1)), namely, the blind spots caused by the offset of the apex of the sensor cone from the center of rotation of the spacecraft. The correctness of Steps 1-12 alone in the absence of this artifact was left unanswered in [1].

B. Optimality bounds

For our purposes, we will consider the algorithm which minimizes the maximum worst-case total angle traversed of any spacecraft S_i to be the optimal algorithm. Other reasonable options would include the algorithm which minimizes the worst-case sum over all spacecraft S_i of the total angle traversed.

In this section, Theorem 3.4 will prove that **Opposing Sensor Constraint** is optimal. Theorem 3.5 shows an equivalence between worst-case bounds for 2 spacecraft and worst-case bounds for any number $n > 2$ of spacecraft. Theorem 3.6 gives a lower bound for the 2-D problem and Theorem 3.7 gives a lower bound on solid angle covered by any algorithm solving the 3-D problem. This bound induces a lower bound on angle traversed in 3 dimensions (see Remark 2.7).

Theorem 3.4: (Justification of the **Opposing Sensor Constraint**): Let S_1 and S_2 be two spacecraft. The most optimal algorithm to guarantee that S_1 and S_2 attain sensor lock is one which uses the **Opposing Sensor Constraint**.

Proof: Imagine there is some algorithm A which achieves sensor lock between S_1 and S_2 in time t_{lock}. Create a new algorithm A^* in which S_1 implements A, but S_2 maintains the **Opposing Sensor Constraint** with S_1. If S_2 had been following A, the apex of $C_{S_2}(t_{lock})$ would be in $C_{S_1}(t_{lock})$ at time t_{lock}. Since S_1 is following A in algorithm A^*, the apex of $C_{S_2}(t_{lock})$ is in $C_{S_1}(t_{lock})$ when both craft follow A^*. By symmetry properties of the **Opposing Sensor Constraint**, the apex of $C_{S_1}(t_{lock})$ is in $C_{S_2}(t_{lock})$, thus guaranteeing sensor lock at or before time t_{lock}. This means that for any algorithm, A, which guarantees sensor lock, a modified algorithm (A^*) which maintains the **Opposing Sensor Constraint** can be constructed such that A^* guarantees sensor lock in at most as much worst-case rotation as A. ∎

Theorem 3.5 (Extending worst-cases to n Spacecraft): Given a spacecraft S_n with sensor cone half-angle Θ_{fov}, and any $\varepsilon > 0$, the worst-case total angle traversed by S_n while performing a correct algorithm with $n-1$ other spacecraft is identical to the worst-case total angle traversed by a spacecraft with sensor cone half-angle $\Theta_{\text{fov}} + \varepsilon$ performing a correct algorithm with one other spacecraft.

Proof: Let t_{worst} be the worst-case time for 2 spacecraft to find each other given a maximum angular velocity of ω_{max}. Clearly the worst-case time for n craft is no worse then this. Pick the initial conditions of the first $n-1$ spacecraft arbitrarily. Let C be the set of communications the first $n-1$ craft would send if they start from these conditions and fail to achieve sensor lock with S_n by time t_{worst}. Let T be the trajectory S_n would take given communications C. Let A_t be the algorithm for two spacecraft, S_1 and S_2, under which each S_1 blindly follows T and S_2 maintains the opposing sensor constraint with respect to S_1. Let P_{worst} and v_{worst} be the initial position and velocity of S_1 with respect to S_2 that achieves the worst-case total angle traversed for S_1 under A_t. In the n spacecraft case, pick some spacecraft S_i. Set the initial position and velocity of S_n with respect to S_i to be λP_{worst} and λv_{worst} for λ such that $\min_{t\in[0,t_{\text{worst}}]}(\|P_{\text{worst}} + v_{\text{worst}} t\|)\lambda > \frac{r_{\text{worst}}}{\sin(\varepsilon)}$. Since $S_1, \cdots, S_{n-1}$ never get more then r_{worst} apart, these spacecraft are contained within a ball of radius r_{worst} centered at S_i. By construction of λ, these craft stay within an angular ball of ε from S_n's point of view, and thus none of these craft achieve sensor lock with S_n before time t_{worst}. ∎

Theorem 3.5 allows the result from Theorem 3.4 to be generalized to any number of spacecraft. In addition, we will use Theorem 3.5 throughout the remainder of the paper to allow us to analyze worst-case total angle bounds by considering the 2 spacecraft case.

Theorem 3.6 (2-D lower bounds on angle traversed): For any algorithm A which solves the 2-D formation initialization problem, and $\Theta_{\text{fov}} < \frac{\pi}{2}$, the worst-case total angle covered by S_1 performing A is 3π.

Proof: For $\Theta_{\text{fov}} < \frac{\pi}{2}$, by Theorem 3.1, every vector, v, on the 2-sphere must be scanned at least once before the final scan of $-v$. This means S_1 must scan at least half the directions on the unit 2-sphere twice for a total angle covered of 3π. ∎

From Theorem 3.6 we can deduce that the worst-case minimum total angle traversed by any correct formation initialization algorithm in 2-D is $min(3\pi - 2\Theta_{\text{fov}}, 4\pi - 4\Theta_{\text{fov}})$.

Theorem 3.7 (3-D lower bounds on solid angle covered): For any algorithm A which solves the 3-D formation initialization problem, and $\Theta_{\text{fov}} < \frac{\pi}{2}$, the worst-case total solid angle covered by S_1 performing A is 6π.

Proof: The total solid angle of a sphere is 4π. For $\Theta_{\text{fov}} < \frac{\pi}{2}$, by Theorem 3.1, every vector, v, on the 3-sphere must be scanned at least once before the final scan of $-v$. This means S_1 must scan at least half the directions on the unit 3-sphere twice for a total solid angle covered of 6π. ∎

Corollary 3.8 (3-D lower bounds on total angle): For any algorithm A which solves the 3-D formation initialization problem, and $\Theta_{\text{fov}} < \frac{\pi}{2}$, the worst-case total angle traversed by S_1 performing A is at least $\frac{3\pi - \alpha_0/2}{\sin\Theta_{\text{fov}}}$ where $\alpha_0 = 2\pi(1-\cos(\Theta_{\text{fov}}))$.

Proof: Recall from Remark 2.7 that $\frac{d}{dt}F_{\text{solid}}(t) = f_{\text{solid}}(\omega) \le 2\|\omega\sin(\Theta_{\text{fov}})\|$. Since $6\pi - \alpha_0 \le \int f_{\text{solid}} dt \le \int 2\|\omega\sin(\Theta_{\text{fov}})\| dt = 2\sin(\Theta_{\text{fov}})\int\|\omega\| dt$ and the total angle rotated is defined as $\int\|\omega\| dt$, we can say that the total angle rotated by any spacecraft S_1 performing A is $\frac{6\pi - \alpha_0}{2\sin(\Theta_{\text{fov}})}$. ∎

IV. Provably correct formation initialization algorithms

Having given lower bounds on what is necessary for a correct formation initialization solution, in this section we set out to answer whether the problem as we pose it has a solution. Section IV-A describes an algorithm from the literature for a 2 dimensional variant of this problem. Section IV-B presents a purely rotational algorithm for formation initialization in 3 dimensions and Theorem 4.6 gives a proof of its correctness. Components of the full 3-D problem will be reduced to the 2-D problem, and the correctness of the 2-D problem will be used in the proof of correctness of the 3-D problem. Section IV-C provides an algorithm which comes closer to the optimality bounds presented in Section III at the expense of other practical considerations. This algorithm is presented largely as a demonstration of the tightness of our optimality bounds.

A. *Formation initialization in two dimensions*

In order to prove the correctness of the algorithm in deep space, we will need a simpler algorithm for the 2 dimensional case, which we term "in-plane search". This algorithm solves the formation initialization problem for a group of spacecraft residing in a plane. Readers should note that the in-plane search algorithm presented here is by [1]. It is described in Table II.

Name: PLANAR SPACECRAFT LOCALIZATION ALGORITHM
Goal: Solve the Formation Initialization problem in 2 dimensions
Assumes: Assumptions in Section II

---

if $S_i \in G_1$ **then**
  Turn to common reference orientation Θ_{start}
else
  Turn to $\Theta_{\text{start}} + \pi$
end if
At synchronized start time t_s, begin rotating with constant angular velocity $\omega > 0$. Continue this rotation for 3π radians.

TABLE II
PLANAR SPACECRAFT LOCALIZATION ALGORITHM.

The next result is proved in [1].

Proposition 4.1: Under Assumptions in Section II, the PLANAR SPACECRAFT LOCALIZATION ALGORITHM achieves formation initialization.

Remark 4.2: PLANAR SPACECRAFT LOCALIZATION ALGORITHM achieves the lower bound from Theorem 3.6. •

B. SPATIAL SPACECRAFT LOCALIZATION ALGORITHM

Both the description of the full 3-D algorithm and its proof of correctness require some additional specific definitions, that we briefly exposed next.

For the purpose of this algorithm, we will define $\Theta_{\text{tilt}} = \min\{\Theta_{\text{sun}}, \Theta_{\text{fov}}\}$ and assume $\Theta_{\text{fov}} \geq \frac{\pi}{4}$.

Definition 4.3: Let S be a spacecraft. Define

- $R_1(S) = \{\vec{u} \in CMF(S) \,:\, \vec{u} \cdot X_S \leq 0\}$;
- $R_2(S) = CMF(S) \setminus R_1(S)$.

Remark 4.4: Let Θ_{tilt} be an angle such that $\frac{\pi}{2} - \Theta_{\text{fov}} < \Theta_{\text{tilt}} < \Theta_{\text{fov}}$. $R_1(S)$ is chosen so as to be included within the region swept out by spacecraft S's sensor cone while it is tilted by an angle Θ_{tilt} towards the sun axis and performing a 3π rotation about the sun axis. $R_2(S)$ is chosen so as to be included within the region swept out by spacecraft S's sensor cone while it is tilted $\frac{\pi}{2} - \Theta_{\text{fov}} < \Theta_{\text{tilt}} < \Theta_{\text{fov}}$ away from the sun axis and performing a 3π rotation about the sun axis. Also, note that in the frame $CMF(S)$, $R_1(S) \cup R_2(S) = \mathbb{R}^3$. •

The full 3-D algorithm will invoke the subroutine described in Table III.

At the end of the execution of 3-D REGION SWEEP ALGORITHM, if S_i is in G_1, then $R_n(S_i)$ has been swept, otherwise S_i has maintained an orientation such that for all S_j in G_1 $M_{S_i}[0,0,1]^T = -M_{S_j}[0,0,1]^T$.

Name: 3-D REGION SWEEP ALGORITHM
Goal: Scan a region for use as a subroutine by SPATIAL SPACECRAFT LOCALIZATION ALGORITHM
Inputs: (i) A spacecraft, S_i
(ii) An integer, $n \in \{1,2\}$, indicating the region to be swept
Assumes: (i) Assumptions in Section II.
(ii) $\Theta_{\text{fov}} \geq \frac{\pi}{4}$ and $\Theta_{\text{fov}} + \Theta_{\text{sun}} \geq \frac{\pi}{2}$.

Require: At the start of this subroutine, there exist matrices $M_1, M_2 \in SO(3)$ such that for all $S_i \in G_1$, $M_{S_i} = M_1$, for all $S_j \in G_2$, $M_{S_j} = M_2$, $M_1[1,0,0]^T = M_2[1,0,0]^T$ and $M_1[0,0,1]^T = -M_2[0,0,1]^T$.
Require: At the start of this subroutine, $[0,0,1]M_1[0,1,0]^T = 0$.
Set $\Theta_{\text{ROT}} = [0,0,1]M_S[0,0,1]^T(-1^n) \cdot \Theta_{\text{tilt}}$
Rotate by Θ_{ROT} about Y_{S_i}
Begin rotating about X_{S_i} by a constant angular velocity ω. Continue this rotation for 3π radians and then stop.
Rotate by Θ_{ROT} about Y_{S_i}

TABLE III
3-D REGION SWEEP ALGORITHM.

We are now ready to define SPATIAL SPACECRAFT LOCALIZATION ALGORITHM (cf. Table IV).

Name: SPATIAL SPACECRAFT LOCALIZATION ALGORITHM
Goal: Solve the Formation Initialization problem in 3 dimensions
Assumes: (i) Assumptions in Section II.
(ii) $\Theta_{\text{fov}} \geq \frac{\pi}{4}$ and $\Theta_{\text{fov}} + \Theta_{\text{sun}} \geq \frac{\pi}{2}$.

---

if $S_i \in G_1$ **then**
  Rotate to align M_{S_i} with I_3
else
  Rotate to align M_{S_i} with M_{opp}
end if
Wait for common start time t_s
Call **3-D REGION SWEEP ALGORITHM** on S_i and $R_1(S_i)$
Call **3-D REGION SWEEP ALGORITHM** on S_i and $R_2(S_i)$
Call **3-D REGION SWEEP ALGORITHM** on S_i and $R_1(S_i)$

TABLE IV
SPATIAL SPACECRAFT LOCALIZATION ALGORITHM.

1) Analysis of SPATIAL SPACECRAFT LOCALIZATION ALGORITHM *:*

Let us discuss the correctness of this algorithm. As in Section IV-A, we reduce the problem to that of two spacecraft finding each other. Call these spacecraft $S_1 \in G_1$ and $S_2 \in G_2$.

Recall that S_2's motion in $CMF(S_1)$ is along a straight line with constant velocity.

Consider the two half-spaces defined by the $\{Y,Z\}$ plane in $CMF(S_1)$. Because S_2 moves with constant velocity with respect to S_1, it can cross from one half-space to the other at most once.

The paths it can take are as follows. S_2 can begin in $R_1(S_1)$ and cross to $R_2(S_1)$ at most once. Likewise S_2 can begin in $R_2(S_1)$ and cross into $R_1(S_1)$ at most once.

Because we make no assumptions about the speed at which these spacecraft take these paths, or at which part of the path they start, handling these cases will automatically handle the cases for paths that fail to cross the $\{Y,Z\}$ plane.

Lemma 4.5 (Partial reduction to in-plane search): Doing a 3π sweep (turning about the sun line) through $R_n(S)$, $n \in \{1,2\}$, $S \in G_1$, finds all spacecraft in G_2 that stay in $R_n(S)$ during the entire duration of the 3π rotation.

Proof: Projecting the centerline of the cone and the spacecraft path onto the $\{Y,Z\}$ plane in $CMF(S)$ reduces this to the 2-D algorithm. In the cases where $R_n(S)$ contains points which project directly onto $(0,0)$ there can be a collision in the 2-D projection which does not correspond to a collision of the craft in 3-D. In these cases, the sensor cone of S_1 always contains all such points, and any colliding craft are found. ■

Finally, we are in a position to establish the correctness of the full 3-D algorithm.

Theorem 4.6: Under Assumptions in Section II, the SPATIAL SPACECRAFT LOCALIZATION ALGORITHM solves the formation initialization problem.

Proof: Consider two spacecraft, S_1 and S_2. Let S_2 start in $R_{begin}(S_1)$ and end in $R_{end}(S_1)$. If $R_{begin}(S_1) = R_{end}(S_1)$ we are done. Otherwise S_1 must scan $R_{end}(S_1)$ at least once after the first scan of $R_{begin}(S_1)$. If the scan of $R_{begin}(S_1)$ did not find S_2, then S_2 must be in $R_{end}(S_1)$

If S_2 never crosses the $\{Y,Z\}$ plane, either the scan of $R_1(S_1)$ or the scan of $R_2(S_1)$ must find it. Otherwise, S_2 starts in one region and ends in the other. The sequence of region sweeps performed by S_1 guarantee that S_1 will scan the region S_2 starts in at least once before scanning the region S_2 ends in. If S_2 is not found when S_1 first performs a sweep of the region in which S_2 begins (call this $R_{begin}(S_1)$), then S_2 must be in the remaining region ($R_{end}(S_1)$) by the end of the sweep. Since this was the first sweep of $R_{begin}(S_1)$, S_1 must scan at $R_{end}(S_1)$ at least once after this point and find S_2. ■

Remark 4.7: SPATIAL SPACECRAFT LOCALIZATION ALGORITHM sweeps a total solid angle of $9\pi + \frac{5\Theta_{\text{tilt}}}{\sin\Theta_{\text{fov}}}$ and performs rotations totaling $9\pi + 5\Theta_{\text{tilt}}$, where $\Theta_{\text{tilt}} := \min(\frac{\pi}{2} - \Theta_{\text{fov}}, \Theta_{\text{sun}})$. •

C. WAIT AND CHECK ALGORITHM

As pointed out in Remark 4.7, the provably correct SPATIAL SPACECRAFT LOCALIZATION ALGORITHM is far from optimal both in terms of total angle traversed and solid angle covered. In what follows, we introduce the WAIT AND CHECK ALGORITHM (cf. Table V). This algorithm has a much better performance with regards to solid angle covered, at the expense of a longer execution time. After we prove its correctness (cf. Theorem 4.9), we show how to modify it to achieve an optimal total rotation given its solid angle covered (cf. Remark 4.10).

The next lemma will be used in establishing the correctness of WAIT AND CHECK ALGORITHM.

Lemma 4.8: Consider a spacecraft S_2 traveling in a path with respect to S_1 with velocity V_{S_2} and point of closest approach $p_{\text{closest}}(S_1,S_2)$. Let $\Pi_{1,2}$ be the plane in $CMF(S_1)$ spanned by the vectors $p_{\text{closest}}(S_1,S_2)$ and V_{S_2}. Define a parameterization of vectors in $\Pi_{1,2}$ by the function $\Theta_{scan}(P) := \arctan(p_{\text{closest}}(S_1,S_2)\cdot P, -V_{S_2}\cdot P)$. For any angles $\Theta \in [0,\pi]$ and $\varepsilon \in [0,\Theta]$, if S_1 first verifies that $\Theta_{scan}(P_{S_2}) < \Theta - \varepsilon$ at time t_1 and then verifies that $\Theta_{scan}(P_{S_2}) > \Theta + \varepsilon$ at time t_2, then by time $t_2 + \frac{\tan(\frac{\pi}{2}-\varepsilon)}{\varepsilon}(t_2 - t_1)$, S_2 will be within ε of its final angle.

Name: WAIT AND CHECK ALGORITHM
Goal: Solve the formation initialization problem using near-optimal solid angle coverage.
Assumes: (i) Assumptions in Section II.
(ii) $\Theta_{\text{fov}} > \frac{\pi}{4}$.

Define $\Theta_\varepsilon = \Theta_{\text{fov}} - \frac{\pi}{4}$
if $S_i \in G_1$ **then**
  Rotate to align M_{S_i} with I_3
else
  Rotate to align M_{S_i} with M_{opp}
end if
Wait for common start time t_s
Rotate by $\frac{\pi}{4}$ about Y_{S_i} {Call this time t_1}
Rotate about X_{S_i} by 2π with angular velocity ω {Call this time t_2}
Wait $\frac{\tan(\frac{\pi}{2}-\Theta_\varepsilon)}{\Theta_\varepsilon}(t_2 - t_1)$ {Call this time t_3}
Rotate about Y_{S_i} by $\frac{-\pi}{2}$ {Call this time t_4}
Rotate about X_{S_i} by 3π with angular velocity ω {Call this time t_5}
Rotate about Y_{S_i} by $\frac{-\pi}{2}$ {Call this time t_6}
Wait $\frac{\tan(\frac{\pi}{2}-\Theta_\varepsilon)}{\Theta_\varepsilon}(t_4 - t_5)$ {Call this time t_1}
Rotate about X_{S_i} by 2π with angular velocity ω {Call this time t_7}

TABLE V
WAIT AND CHECK ALGORITHM.

Proof: Since $\Theta_{scan}(P_{S_2}(t2)) - \Theta_{scan}(P_{S_2}(t1)) > 2\varepsilon$, $\frac{\|V_{S_2}\|}{\|p_{\text{closest}}(S_1,S_2)\|}$ is at least $\frac{2\varepsilon}{t_2-t_1}$. $\Theta_{scan}(P_{S_2}(\frac{\tan(\frac{\pi}{2}-\varepsilon)}{\varepsilon}(t_2 - t_1))) \geq \arctan(\tan(\Theta+\varepsilon) + \frac{2\varepsilon}{t_2-t_1}\frac{\tan(\frac{\pi}{2}-\varepsilon)}{\varepsilon}(t_2-t_1)) \geq \pi - \varepsilon$. ■

Next, we characterize the correctness of the WAIT AND CHECK ALGORITHM.

Theorem 4.9: The WAIT AND CHECK ALGORITHM correctly solves the formation initialization problem.

Proof: Consider a spacecraft, S_1. Any other spacecraft whose X position is less than zero at time t_1 must either be found, cross the $\{Y,Z\}$ plane, or cross the $\{X,Z\}$ plane before t_2. If S_2 crossed the $\{X,Z\}$ plane between t_1 and t_2 and was not found, then it must have been moving with sufficient velocity to have moved to within Θ_ε of its final angle by time t_3. By this logic, by t_3, any craft with a final angle corresponding to a positive X component of position must have been found by time t_2, or be on the $+X$ side of the $\{Y,Z\}$ plane by time t_3. Between t_4 and t_5 all such craft are found, along with any craft that started on the $+X$ side of the $\{Y,Z\}$ plane and have not left it by t_5 (by Lemma 4.8). Any craft which have left the $+X$ side of the $\{Y,Z\}$ plane by t_5 but were not found during the sweep of the $-X$ half of the $\{Y,Z\}$ plane must have been moving with sufficient angular velocity as to be within Θ_ε of their final angles (on the $-X$ half of the $\{Y,Z\}$ plane) by t_6 (cf. Lemma 4.8). For this reason, the final sweep of the $-X$ side of the $\{Y,Z\}$ plane need only be a 2π sweep. ■

Remark 4.10 (Angle-optimal region sweeps): The WAIT AND CHECK ALGORITHM covers a solid angle of $7\pi + \frac{5\Theta_{\text{tilt}}}{\sin(\Theta_{\text{tilt}})}$. Clearly, the ratio of total angle traversed to solid angle covered in WAIT AND CHECK ALGORITHM is not at the optimal $\frac{1}{2\sin(\Theta_{\text{fov}})}$. The algorithm can be modified to traverse a total angle of $7\pi\sin(\Theta_{\text{tilt}}) + 5\Theta_{\text{tilt}}$, where $\Theta_{\text{tilt}} := \min(\pi/2 - \Theta_{\text{fov}}, \Theta_{\text{sun}}, \Theta_{\text{fov}})$, at the expense of not respecting the sun-angle constraint. We describe how next. The optimal ratio of total angle traversed to solid angle covered is achievable for any rotational trajectory of $\vec{v}_{\text{SENSOR}}(S)$ over time. While a rotational velocity, ω, specifies the instantaneous rotation of the entire body frame of S, the instantaneous motion of $\vec{v}_{\text{SENSOR}}(S)$ only fixes two degrees of freedom of this rotation. By choosing ω to lie along $\vec{v}_{\text{SENSOR}}(S) \times \frac{d}{dt}\vec{v}_{\text{SENSOR}}(S)$, we can always achieve the maximum instantaneous $f_{\text{solid}}(\omega)/\|\omega\|$.

Let us suppose that $\vec{v}_{\text{SENSOR}}(S)$ is within an angle of $\frac{\pi}{2} - \alpha$ of the sun line, and we wish for $\vec{v}_{\text{SENSOR}}(S)$ to sweep out the arc defined by $C_\alpha := \{\vec{v} \in \mathbb{R}^3 \ : \ \|\vec{v}\| = 1 \wedge \arccos(\vec{v} \cdot \vec{v}_{\text{SUN}}) = \frac{\pi}{2} - \alpha\}$. At any instant during which $\vec{v}_{\text{SENSOR}}(S) \in C_\alpha$, the optimal axis of rotation, ω, is both perpendicular to $\vec{v}_{\text{SENSOR}}(S)$ and guarantees $\vec{v}_{\text{SENSOR}}(S)$ remains in C_α. One such ω always lies on a cone which we will define as $C_{\text{tumble}} := \{\vec{v} \in \mathbb{R}^3 \ : \ \|\vec{v}\| = 1 \wedge \arccos(\vec{v} \cdot \vec{v}_{\text{SUN}}) = \alpha\}$, see Figure 3. Note that the body

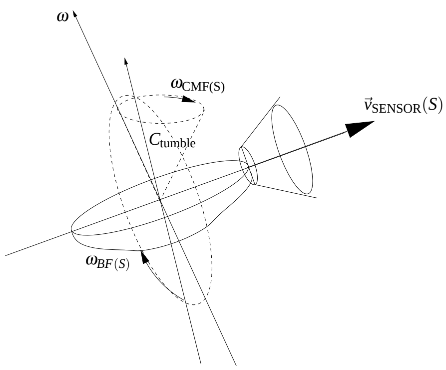

Fig. 3. Performing a sweep of 2π with less then 2π rotation

frame, $BF(S)$ does not move with respect to $CMF(S)$ at any point along the axis ω. When the sweeps about the sun line of WAIT AND CHECK ALGORITHM are executed as we just described, the algorithm requires a total angular rotation of $\frac{7\pi}{\sqrt{2}} + 5\Theta_{\text{tilt}}$. •

V. CONCLUSIONS AND FUTURE WORK

We have considered the formation initialization problem for a group of spacecraft endowed with limited field-of-view relative position sensors and omnidirectional communication. We have obtained optimality bounds for the performance of any correct algorithm in terms of worst-case solid angle covered and total angle traversed. In two dimensions, the angle traversed bound is hard and in three dimensions, the angle traversed bound is no worse than the solid angle bound. Our analysis of optimality justifies several decisions made in both our own algorithm designs and those of previous works, including the PLANAR SPACECRAFT LOCALIZATION ALGORITHM and the **Opposing Sensor Constraint**. We have also synthesized two provably correct formation initialization algorithms. In particular, the SPATIAL SPACECRAFT LOCALIZATION ALGORITHM is simple and easily provable, while the WAIT AND CHECK ALGORITHM is nearly optimal according to the optimality bounds obtained.

Areas of future work include: (i) the determination of the optimality of **Opposing Sensor Constraint** when the spacecraft start in random orientations (this is easily seen for the case of two spacecraft). If this is true in general, then it will be of interest to determine the optimal way to move the spacecraft to satisfy the **Opposing Sensor Constraint**; (ii) the investigation of other notions of optimality, such as minimum time to complete formation initialization on a fixed fuel budget; (iii) the determination of whether the 6π solid angle bound in three dimensions is a hard bound. For $\Theta_{\text{fov}} = \frac{\pi}{2}$, this bound gives a total angle rotated bound of 3π, which matches the intuitive result from reducing this special subproblem to two dimensions.

ACKNOWLEDGMENTS

The authors wish to thank Prof. Roy Smith for pointing our attention to the algorithms described in [1]. This research was supported in part by NASA University Aligned Research Program Award TO.030.MM.D.

REFERENCES

[1] S. R. Ploen, D. P. Scharf, F. Y. Hadaegh, and M. Bikdash, "Initialization of distributed spacecraft formations," *Journal of the Astronautical Sciences*, vol. 52, no. 4, 2004.

[2] H. Hemmati, W. Farr, B. Liu, J. Kovalik, M. Wright, and J. Neal, "Formation metrology," Nov. 2003, high-level description of sensors developed at JPL for precision gormation control. [Online]. Available: http://dst.jpl.nasa.gov/metrology/index.htm

[3] M. Tillerson, G. Inalhan, and J. How, "Coordination and control of distributed spacecraft systems using convex optimization techniques," *International Journal on Robust and Nonlinear Control*, vol. 12, pp. 207–242, 2002.

[4] P. R. Lawson, S. C. Unwin, and C. A. Beichman, "Precursor science for the terrestrial planet finder," Jet Propulsion Laboratory Publication 04-14, Oct. 2004. [Online]. Available: http://planetquest.jpl.nasa.gov/documents/RdMp273.pdf

[5] F. Bauer, J. Bristow, D. Folta, K. Hartman, D. Quinn, and J. How, "Satellite formation flying using an innovative autonomous control system (autocon) environment," in *AIAA Conf. on Guidance, Navigation and Control*, Reston, VA, 1997, pp. 657–666.

[6] A. Das and R. Cobb, "Techsat 21 - space missions using collaborating constellations of satellites," in *AIAA Conf. on Small Satellites*, Logan, UT, 1998.

[7] J. How, R. Twiggs, D. Weidow, K. Hartman, and F. Bauer, "Orion - a low cost demonstration of formation flying in space using GPS," in *AIAA/AAS Astrodynamics Specialist Conf. and Exhibit*, Reston, VA, 1998, pp. 276–286.

[8] J. A. Dooley and P. R. Lawson, "Technology plan for the terrestrial planet finder coronagraph," Jet Propulsion Laboratory Publication 05-8, Mar. 2005. [Online]. Available: http://planetquest.jpl.nasa.gov/TPF/TPF-CTechPlan.pdf

[9] D. P. Scharf, F. Y. Hadaegh, and S. R. Ploen, "A survey of spacecraft formation flying guidance and control (part ii): control," in *American Control Conference*, Boston, MA, June 2004, pp. 2976–2985.

[10] V. Kapila, A. G. Sparks, J. M. Buffington, and Q. Yan, "Spacecraft formation flying: dynamics and control," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 23, pp. 561–564, 2000.

[11] M. Mesbahi and F. Y. Hadaegh, "Formation flying control of multiple spacecraft via graphs, matrix inequalities, and switching," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 24, no. 2, pp. 369–377, 2001.

[12] R. W. Beard, J. Lawton, and F. Y. Hadaegh, "A coordination architecture for spacecraft formation control," *IEEE Transactions on Control Systems Technology*, vol. 9, no. 6, pp. 777–790, 2001.

[13] I. I. Hussein, "Motion planning for multi-spacecraft interferometric imaging systems," Ph.D. dissertation, University of Michigan, 2005.

[14] N. A. Lynch, *Distributed Algorithms*. San Mateo, CA: Morgan Kaufmann Publishers, 1997.

[15] J. E. Marsden and T. S. Ratiu, *Introduction to Mechanics and Symmetry*, 2nd ed. New York: Springer Verlag, 1999.