Inter-Robot Interactions in Multi-Robot Systems Using Braids

Yancy Diaz-Mercado and Magnus Egerstedt

Abstract—This paper describes a framework for multi-robot coordination and motion planning with emphasis on inter-agent interactions. We focus on the characterization of inter-agent interactions with sufficient level of abstraction so as to allow for the enforcement of desired interaction patterns in a provably safe (i.e., collision-free) manner, e.g., for achieving rich movement patterns in a shared space, or to exchange sensor information. We propose to specify interaction patterns through elements of the so-called braid group. This allows us to not focus on a particular pattern per se, but rather on the problem of being able to execute a whole class of patterns. The result from such a construction is a hybrid system driven by symbolic inputs that must be mapped onto actual paths that both realize the desired interaction levels and remain safe in the sense that collisions are avoided.

I. INTRODUCTION

Many applications have been proposed for multi-robot systems. For example, the multi-robot foraging paradigm, in which agents wander around an environment searching for items of interest [1], [2], can relate to many real-world problems (e.g., waste or specimen collection in hazardous environments, explosive detection and disposal, and search and rescue operations after a crisis such as floods and earthquakes). Another application of interest in the research community and in industry is cooperative assembly [3], [4] and self-assembly [5], [6]. In the cooperative assembly scenario, multiple robots need to coordinate their motion in order to cooperatively assemble a structure using a possibly heterogeneous set of tools, end effectors, skills, or parts. Similarly, in the self-assembly scenario, multiple robots need to coordinate their motion in order to collectively form a structure or achieve a geometric configuration. There has been a recent push for robotic farming and precision agriculture [7], [8], where a fleet of robots is sent to gather data on the status of crops, tend to and harvest them. In some communication architectures, mobile agents called data MULEs or message ferries [9]–[11] are used to transport data between sensors, access points, or base stations, in situations where it is impractical to establish physical communication. Multi-robot simultaneous localization and mapping (SLAM) takes advantage of the robot team’s size to attain a more complete (or more detailed) map faster than with a single robot by combining information from multiple sources and coordinating motion [12], [13]. Other applications include transportation systems (e.g., intelligent highways, air traffic control) [14], and the convoy protection scenario (e.g., surveillance, coverage) [15].

In many of these applications, the overall objectives can be stated in terms of making a team of robots follow a physical path, such as a road or the movements of a ground convoy, while ensuring that particular search patterns are executed [16]–[18]. These patterns should be selected in such a way that certain secondary geometric objectives are met, including ensuring that an area along the path is covered, that multiple views of the same objects are achieved, that an aerial vehicle is always on top of the convoy, or that sufficiently many vehicle-to-vehicle interactions take place for the purpose of information sharing [19], [20]. In this work, we collect all of these different secondary objectives under one unified banner, namely multi-robot mixing. In particular, we specify interaction patterns with certain desired levels of mixing, and then proceed to generate the actual cooperative movements that realize these mixing levels.

In this paper we study the problem of characterizing inter-robot interactions for the sake of coordination and collision avoidance. We specify the mixing patterns through elements of the so-called braid group [21], [22], where each element corresponds to a particular mixing pattern, i.e., we do not focus on a particular pattern per se, but rather on the problem of being able to execute a whole class of patterns. The result from such a construction is a hybrid system driven by symbolic inputs [23], i.e., the braids, that must be mapped onto actual paths that both obtain the mixing level specified through the braid, and remain safe in the sense that collisions are avoided.

The use of symbolic inputs allows us to abstract away the geometry and physical constraints involved in the multi-robot motion planning. As described in [23], this provides the advantage of hierarchical abstractions that are typically broken into three stages. At the top layer is the specification level, which describes the motion tasks (such as robots A and B should interact, and arrive at the goal simultaneously). The second layer is the execution level, which describes how to obtain the motion plans, e.g., by generating trajectories based on optimality conditions. The bottom layer is the implementation level, which concerns itself with constructing the robot controller, e.g., to track a reference trajectory. This work also extends the notions presented in [24] for trajectory tracking of virtual vehicles. The idea there is to have the physical vehicles track a virtual vehicle, as opposed to a path (or trajectory) itself. The virtual vehicle, which is being controlled directly with simplified dynamics, can be controlled in order to track a reference path while satisfying the constraints, compensating for the physical vehicle’s dynamics and other disturbances.

The outline of the paper is as follows: in Section II we start with a brief summary of the existing literature on braid-
ing for the sake of multi-robot motion planning; in Section II-B the braid group is introduced as a way of specifying mixing levels and the corresponding symbolic braid objects are given a geometric interpretation in terms of planar robot paths; controllers are then proposed so that mixing strategies satisfying a given specification can be executed by a class of robots as described in Section III, together with a bound on the satisfiability of a given specification can be executed by a class of robots. Implementation of these controllers on actual robotic platforms is addressed, and these ideas are deployed on a team of actual mobile robots.

II. BRAIDS AND ROBOTICS

The use of braids for robot motion planning has been considered before. Using the notion of configuration space for robot motion planning, [25] studies the problem of construction and classification of configuration spaces for paths; e.g., robots on a manufacturing floor constrained on rails or paths. By studying the topological data associated with these graphs, such as the braid groups, he is able to provide a measure of the complexity of the control problem (e.g., the construction of potential field controllers on homeomorphic spaces).

In [27], the graph braid group is used as the fundamental group of the configuration space of graphs that describe robot motion. There, each graph in the configuration space represents discretized collision-free robotic motion plan (e.g. road maps), where at each discrete time the graph vertices represent the positions of robots and (possibly moving) obstacles, and the edges represent fixed tracks connecting these vertices. They present an algorithm to construct the presentation of the graph braid group of \( N \) agents, where the group generators (i.e., the braids) represent actual paths between configurations of robots on the graph, i.e., the motion plan to transition from one configuration to another. However, they only consider zero-size robots where they rely on a “one edge separation” between points at all times to avoid collisions, and as such, their approach is purely academic and mainly focuses on the combinatorics of ideal robots moving on a graph.

We will use the braid group’s generators as symbolic inputs to multi-robot hybrid controllers which characterize and enforce collision-free interactions, take into account kinematic and geometric constraints, and are executable on actual robotic platforms. This is the first approach that uses braids in multi-robot motion planning to symbolically characterize and enforce rich interaction patterns, implementable on actual robotic platforms. In particular, we extend the notions presented in [28], [29]. The definition on planar braids and their geometric interpretations are generalized from their original form presented in [28]. A controller was designed for nonholonomic vehicles to track a geometric path in [29]. In this paper, we modify this controller to instead produce optimal trajectories that are provably safe in the collision-free sense, satisfy a set of spatio-temporal constraints, and follow desired geometric paths (Section II-B). This also allows us to obtain tighter bounds on the amount of interaction patterns that are achievable in a space than the one found in [29]. Two new contributions to this work are found in Section III-C where these trajectory generating controllers are extended to non-rectangular regions, and Section IV-A which includes development of an optimal trajectory tracking controller with formal guarantees on optimality and spatio-temporal constraint satisfaction.

A. Planar Braids

Consider two agents on a square, initially located at the two left vertices of the square as in Fig. 1a. The agents’ task is to move to the two right vertices of the square. There are two ways in which these target vertices can be assigned. The first is to simply let the robots move along a straight line while the second is to have them cross paths and move to vertices diagonally across from their initial placement. If the robots are not to collide with each other, one agent can cross the intersection of the two paths first, and then the other (or vice versa). In the braid group, these two options correspond to different “braids,” and we have thus identified three planar braids for two agents, as shown in Fig. 1b. Let us momentarily denote these three braids, \( \sigma_0, \sigma_1, \sigma_1 \).

Now, given these three braids, we can concatenate them together to form other braids, as seen in Fig. 2. The left braid is given by \( \sigma_1 \cdot \sigma_1 \) and the right braid is \( \sigma_1 \cdot \sigma_1 \). In the braid group, what really matters is not the geometric layout of the paths, but how the paths wrap around each other. As can be seen, if we were to “pull” the right corners in \( \sigma_1 \cdot \sigma_1 \), the “strands” would get “tangled” in the middle, while a “stretched-out” \( \sigma_1 \cdot \sigma_1 \) is simply \( \sigma_0 \). Thus we let \( \sigma_0 \) be the identity braid, such that \( \sigma_1 \) and \( \sigma_1 \) are each others’ inverses in the sense that

\[
\sigma_1 \cdot \sigma_1 = \sigma_1 \cdot \sigma_1 = \sigma_0.
\]

In fact, every braid has an inverse and, as such, the set of braids (together with the concatenation operation) is indeed a group. And, as \( \sigma_1^{-1} = \sigma_1 \) and \( \sigma_1^{-1} = \sigma_1 \), \( \sigma_0 \) and \( \sigma_1 \) (or \( \sigma_0 \) and \( \sigma_1 \) for that matter) are the so-called generator braids for this group in that all planar braids can be written as concatenations of these two braids and their inverses [21], [22].

This notion of generator braids can be extended to the case when there are \( N \geq 2 \), with the only difference being that we will have \( N \) generators rather than just two, i.e., let \( \sigma_0 \) be the trivial generator with no interactions and \( \sigma_k \) denote the interactions between agents \( k \) and \( k + 1, k = 1, \ldots, N - 1 \). If we let \( \Sigma_N \) be the set of all planar generator braids over \( N \) agents, this set will serve as the alphabet over which braid strings (themselves braids) are produced from, and we let \( \Sigma_N^M \) denote the set of all braids of length \( M \) (i.e., braids composed of \( M \) generators) over \( N \) agents.
sets of intermediary points are uniformly spaced in $h$.

More on this in Section III-C.

B. A Geometric Interpretation

Although heavily inspired by geometry, the braid group is not concerned with the actual geometry of the braid strands. For the sake of describing the robot motion plans, we will associate geometric paths with the different braids. First of all, we assume that the braid is geometrically located in a rectangular area of height $h$ and length $\ell$ no matter how long the braid string is. Using the particular two-agent braids discussed in the previous paragraphs, we assume that the three agents are initially located at the points (0, 0) and (0, h), while the final locations are at $\ell$ and $\ell$. If the total braid results from the use of one single generator braid $\sigma_0$, or $\sigma_1$ then no additional points are needed. However, if the braid has length 2, then we also need to introduce intermediary “half-way” points $(\ell/2, 0)$ and $(\ell/2, h)$. As such, we let $\mathcal{P}_0 = \{(\ell/4, 0), (\ell/4, h)\}$ be a set of uniformly spaced positions for these intermediary points, where the subscript 2 denotes the two-agent case, $M$ is the length of the braid to be executed, and $q = 0, 1, \ldots, M$.

Using this notation, we can refer back to Fig. 1a and say that each of the two generator braids correspond to an assignment, i.e., a bijective map, between $\mathcal{P}_0$ and $\mathcal{P}_1$, and we use the following notation to denote this fact

$$\sigma_i : \mathcal{P}_0 \rightarrow \mathcal{P}_1, \ i = 0, 1,$$

where $\rightarrow$ denotes “bijection”. Note that this is a slight abuse of notation in that $\sigma_i$ now denotes both an element in the braid group as well as a map – this distinction, however, should be entirely clear from the context. Further, we will refer to these points which agents are bijectively mapped to and from as braid points.

If we generalize this to $N \geq 2$ agents and let $\sigma$ denote a string of generators of length $M \geq 2$, i.e., $\sigma \in \Sigma_N^M$, we will use the notation

$$\sigma(k) : \mathcal{P}_0^{(k-1)} \rightarrow \mathcal{P}_1^k, \ k = 1, \ldots, M,$$

where $\mathcal{P}_i$ is a string of braids comprised of $i$ braids.

Fig. 3. The geometric interpretation of braid string $\sigma = \sigma_2 \cdot \sigma_1$ for the three-agent case. In this example, $\sigma(1) = \sigma_2$ and maps $\mathcal{P}_0^2$ to $\mathcal{P}_1^2$, while $\sigma(2) = \sigma_1$ and maps $\mathcal{P}_0^1$ to $\mathcal{P}_1^1$.

where $\sigma(k)$ is the $k^{th}$ braid in the string $\sigma$ and

$$\mathcal{P}_N^q = \{ (\frac{q}{M} \ell, 0), (\frac{q}{M} \ell, \frac{1}{N-1} h), (\frac{q}{M} \ell, \frac{2}{N-1} h), \ldots, (\frac{q}{M} \ell, h) \},$$

as shown for the three-agent case in Fig. 3.

We moreover use the notation $\xi(i, j) \in \mathbb{R}^2$ to denote the point agent $j$ should go to at step $i$, $i = 1, \ldots, M$. We use the convention that $\xi(0,j) = (0, (j-1)h/(N-1))$, $j = 1, 2, \ldots, N$, to denote agent $j$’s initial position. In other words,

$$\xi(1,j) = \sigma(1) \langle \xi(0,j) \rangle,$$

$$\xi(2,j) = \sigma(2) \langle \xi(1,j) \rangle = \sigma(2) \circ \sigma(1) \langle \xi(0,j) \rangle,$$

or more generally,

$$\xi(i,j) = \sigma(i) \langle \xi(i-1,j) \rangle = \sigma(i) \circ \sigma(i-1) \circ \cdots \circ \sigma(1) \langle \xi(0,j) \rangle,$$

where we use the $\langle \cdot \rangle$ notation to denote the argument to $\sigma(\cdot)$ and $\circ$ to denote composition. This construction is also illustrated in Fig. 3 for the three-agent case.

The geometric interpretation we will make of the planar braids is that the mobile agents that are to execute them must traverse through these braid points. They must moreover do so in an orderly and safe manner, which will be the topic of the next section.

III. Executing Braids

Given a collection of $N$ agents with dynamics

$$\dot{x}_j = f(x_j, u_j),$$

and planar output

$$y_j = h(x_j) \in \mathbb{R}^2, \ j = 1, \ldots, N,$$

we will denote the mobile agents that are to execute them as the application demands. More on this in Section III-C.

Fig. 2. Two concatenated braids $\sigma_1 \cdot \sigma_1$ and $\sigma_1 \cdot \sigma_1$$1$Unless otherwise stated, the rest of this document we will assume these sets of intermediary points are uniformly spaced in $h$ and $\ell$, but the notions presented here extend to points which are not, as the application demands.

2Note that a “braid” here refers to a member of the braid group, e.g., a single generator (viz., a single bijective map) or a concatenation of several generators (viz., a composition of bijective maps).
then it is of interest to have these agents execute a braid 
\( \sigma \in \Sigma^M_N \). We now define what it means for this braid to be 
executed.

Given an input braid string \( \sigma \), what each individual agent 
should do is “hit” the intermediary braid points \( \xi(i, j) \) at 
specified time instances. We let \( T \) denote the time \( \sigma \) should 
take for the entire string to be executed. As such, the first 
condition for a multi-agent motion to be feasible with respect 
to the braid is the following:

**Definition 3.1 (Braid-Point Feasibility):**
A multi-robot trajectory is braid-point feasible if
\[
y_j(t_i) = \xi(i, j), \quad i = 0, \ldots, M, \quad j = 1, \ldots, N.
\]
where the \( t_i \)'s form a partition of a given time window \( [0, T] \), 
i.e.,
\[
t_0 < t_1 < \cdots < t_i < \cdots < t_M = T.
\]

On top of braid-point feasibility, we also insist on the robots 
not colliding during the maneuvers. To a certain degree, this 
condition is what restricts the level of mixing that is possible, 
i.e., since the braid is constrained in a rectangle of fixed height 
and width, what length stretches the multi-robot system can 
execute while maintaining a desired level of safety separation.

**Definition 3.2 (Collision-Free):**
A multi-robot trajectory is collision-free if
\[
\|y_i(t) - y_j(t)\| \geq \delta_{ij}, \quad \forall i \neq j, \quad t \in [0, T],
\]
where \( \delta_{ij} > 0 \) is the desired level of safety separation between 
agents \( i \) and \( j \).

For convenience, we will refer to \( \delta = \max \delta_{ij} \) as the 
maximum safety separation such that no agent collides. This 
will come up in Theorem \( \square \). We are missing a notion to 
describe what the multi-robot mixing problem is, that is, what 
institutes a braid controller.

**Definition 3.3 (Braid Controller):**
A multi-robot controller is a braid controller if the resulting 
trajectories are both braid-point feasible and collision free, 
for all collision-free initial conditions such that
\[
y_j(0) = \xi(0, j), \quad j = 1, \ldots, N.
\]

As a final notion, we are interested in how much mixing a 
particular system can support.

**Definition 3.4 (Mixing Limit):**
The mixing limit \( M^* \) is the largest integer \( M \) such that there 
exists a braid controller for every string in \( \Sigma^M_N \).

Whenever two strands of the braid associated with a given 
braid string cross, the two associated agents will have to inter-
act. The mixing limit therefore serves as an input-independent 
bound on how much mixing is achievable for a given team of 
agents operating in a given environment. The mixing limit is 
in general quite hard to compute, it needs to consider every 
permutation of strings of varying lengths up to some number, 
the geometry assigned to each string and is dependent on the 
kinematical response of the multi-robot system. However, 
under certain assumptions it is possible to find bounds on \( M^* \) 
for a given braid controller.

### A. Braid Controllers: Stop-Go-Stop Hybrid Strategy

Our first attempt at executing braids will be a hybrid control 
strategy called the Stop-Go-Stop. We will assume that agent 
dynamics are given by single integrator dynamics, i.e., \( \dot{x}_j = u_j \in \mathbb{R}^2 \) with \( y_j = x_j, \quad j = 1, \ldots, N \). Further, for practical 
considerations, assume that there is a cap on the maximum 
velocity achievable by the agents, i.e., \( \|u_j\| \in [0, v_{\text{max}}] \).
The idea is then at each braid step to “send” agents off straight 
to their next braid point in order of the distance they’ll need 
to travel, waiting just long enough to avoid collisions before 
sending an agent off. To that end, we define \( s_i : \{1, \ldots, N\} \rightarrow 
\{1, \ldots, N\} \) to be a bijective mapping that denotes the farthest 
distance ordering at step \( i \), that is
\[
s_i(p) < s_i(q) \implies \|\xi(i, p) - \xi(i - 1, p)\| \geq \|\xi(i, q) - \xi(i - 1, q)\|
\]
where ties are arbitrarily broken such that \( s_i \) remains a 
bijection. We let the agents heading angle be given by \( \theta_{i,j} = \tan^{-1} \left( \frac{\xi(i, j) - \xi(i - 1, j)}{\xi(i, j) - \xi(i - 1, j)} \right) \) where the subscript indicates the 
first or second component, and the unit heading vector be \( \hat{\rho}_{i,j} = [\cos(\theta_{i,j}), \sin(\theta_{i,j})]^T \). Once a braid sequence 
exists, we will assume that agent \( i \) starts at \( s_i \).

**Theorem 1:** The STOP-GO-STOP controller in Fig. 4 is a 
braid controller if the braid points themselves are sufficiently 
separated and
\[
\cos(\theta^*) v_{\text{max}} \left( \min_i (t_i - t_{i-1}) - (N - 1)\tau \right) \geq \sqrt{\ell^2 / M^2 + h^2}.
\]

**Proof:**
The STOP-GO-STOP controller ensures that the agents are 
ever within \( \delta \) of each other by virtue of the fact that they 
have to wait until they are indeed at least that far apart 
(horizontally) before entering GO mode. As such, the trajectories 
are collision-free.

What remains to show is that they are also braid-point 
feasible. Consider the agent that has to wait the longest before 
it can move, i.e., the agent that has to wait a total of \( (N - 1)\tau \), 
and at its worst has \( \min_i (t_i - t_{i-1}) - (N - 1)\tau \) time left 
to reach the next braid point. In other words, we need that 
the distance traveled in that amount of time at the speed
\footnote{Since we are interested in the mixing limit, the analysis is done with the 
horizontal direction being the limiting factor for safe execution of the braids.}
words, we need the largest distance possible (which it really is not). In other
and, as we are only looking for a bound, we assume that we
required. But, we note that
and the proof follows.

$$v_{\text{max}} \cos(\theta_{i,s-1}(1)) / \cos(\theta_{i,s}(j))$$ is greater than the distance
and, as we are only looking for a bound, we assume that we
use this lower speed and that the distance required to travel is
the largest distance possible (which it really is not). In other
words, we need

$$\cos(\theta^*) v_{\text{max}} \left( \min_{i} (t_i - t_{i-1}) - (N - 1) \right) \geq \sqrt{t^2/M^2 + h^2},$$

and the proof follows.

Note that Theorem 1 implicitly provides a lower bound on the
mixing limit as long as the agents’ paths are straight lines.
In Fig. 3 we see the mixing limit as a function of the
number of agents in the team for parameters $v_{\text{max}} = 5$, $T = 20$, $\ell = 5$, $h = 10$, and $\Delta = 0.2$. The problem with this
strategy is that it does not allow for more general geometric
paths (which could be interpreted as feasible trajectories under
a given robot dynamical model), nor does it ensure that agents
get within a specified distance from each other (which could
be necessary when collaboration is required). The next section
we present a new strategy called braid reparameterization
which will explicitly consider more general strand geometries,
and allow us to obtain analytical bounds on the mixing limit.

B. Braid Controller: Braid Reparameterization

We are now seeking a strategy that will allow us to follow
a given geometry while achieving a mixing strategy encoded
as a braid string $\sigma \in \Sigma_{N}^{M}$. Further, we wish to enforce inter-
agent interaction as dictated in $\sigma$ by having agents get as close
as the safety separation $\delta_{jk}$. Before moving forward, consider
the following lemma.

Lemma 1: If at any braid step, the generators in the braid
substring $\sigma(k) = \sigma \subseteq \Sigma_{N}^{M}$, $m \leq M$, have indices that are two
or more apart, then any agent interacts with at most one other
agent at this step.

Proof:
Let $B_{N}^{k} \in \mathbb{R}^{N \times 2}$ be a matrix that contains the set of
braid points at time $k$ such that $B_{N}^{k} = [\xi(0,1) \cdots \xi(0, N)]^{T}$. Consider the two-generator concatenation $\sigma_{i} \cdot \sigma_{j}$. As a bijective
map, if $\sigma_{i} = \sigma_{0}$, then the agents do not interact, and the agent
in braid point position $[B_{N}^{k-1}^{i}]_{n}$ gets mapped to braid point
position $[B_{N}^{k}]_{n}$, where $[B]_{n}$ corresponds to the $n^{th}$ row of $B$,
$n = 1, \ldots, N$. If $\sigma_{i} \neq \sigma_{0}$, then $\sigma_{i}$ will swap the position of the
two agents occupying the braid point positions $i$ and $i + 1$ at
step $k - 1$, i.e., it maps the agent occupying $[B_{N}^{k-1}^{i}]$ to $[B_{N}^{k}^{i+1}]$
and the agent occupying $[B_{N}^{k-1}^{i+1}]$ to $[B_{N}^{k}^{i}]$. Similarly, $\sigma_{j}$
swaps the position of the agents occupying the braid point
positions $j$ and $j + 1$.

The two agents in positions $i$ and $i + 1$ at $k - 1$ would only
interact with the two agents in positions $j$ and $j + 1$ at $k - 1$
if $i$ or $i + 1$ for that matter) is equal to either $j$ or $j + 1$. But
if we let $|i - j| \geq 2$, then we get that

$$|i - j| \geq 2 \iff 2 \leq i - j \text{ or } 2 \leq j - i$$

$$\Rightarrow j < j + 1 < j + 2 \leq i < i + 1$$
or

$\Rightarrow i < i + 1 < i + 2 \leq j < j + 1$

$$\begin{cases}
  i \neq j \\
  i \neq j + 1 \\
  i + 1 \neq j \\
  i + 1 \neq j + 1,
\end{cases}$$

and as such these two-generator concatenation maps the agents
from one set of braid points to the next with at most two
interaction between at most two agents per interaction.

One of the two braid group relations $\{21\}, \{22\}$ tells us that
if $|i - j| \geq 2$ then $\sigma_{i} \cdot \sigma_{j} = \sigma_{j} \cdot \sigma_{i}$. More generally, if we let

$$h : \{1, \ldots, m\} \rightarrow \{1, \ldots, N - 1\}$$

be a surjective map such that $|h(i) - h(j)| \geq 2$ for all
$i, j \in \{1, \ldots, m\}$ with $i \neq j$, then for any bijective map
$g : \{1, \ldots, m\} \rightarrow \{1, \ldots, m\}$ we have that

$$\sigma(k) = \sigma_{h(1)} \cdot \sigma_{h(2)} \cdots \sigma_{h(m)}$$

As such, the braid generators can be rearranged to obtain any
permutation of two-generator concatenations from generators
in $\sigma(k)$. Since for all permutations of two-generator concate-
inations we will have indices that are two or more apart, these
in order to safely execute braids, one need only consider the case when two agents interact. We will devise a strategy for reparameterizing the geometry such that should two agents interact, the resulting parameters are at least δ from each other. By tracking the parameterized paths, the agents’ trajectories will be collision-free and braid point feasible. The resulting controllers can be combined to satisfy a given braid string.

Consider the geometric path agent \( j \) must follow at step \( i \) given by \( \gamma^j_i : [0, 1] \rightarrow \mathbb{R}^2 \) with \( \gamma^j_i(0) = \xi(i-1, j) \) and \( \gamma^j_i(1) = \xi(i, j) \). Let \( \Delta \) be the arclength of this path, given by

\[
\Delta = \int_0^1 \sqrt{\left(\gamma_i^j(p)\right)^T \gamma_i^j(p)} \, dp.
\]

We wish to find a parameterization of the strand geometries such that the parameters of two intersecting strands, thought of as a virtual vehicles that live on their geometries, are collision-free. The strategy to do so will be to impose constraints on the agents separation from the path intersection at a specified time. To ensure braid-point feasibility, the strategy will be to impose constraints on the time in which the agents must get from the beginning of one braid step to the end of that step. To that end, we propose the following constrained optimization problem

\[
(v^*_j, v^*_k) = \arg \min_{(v_j, v_k)} J(v_j, v_k) = \arg \min_{(v_j, v_k)} \frac{1}{2} \int_{t_{i-1}}^{t_i} v_j^2 + v_k^2 \, d\tau
\]

subject to

\[
\hat{p}_j = v_j, \quad p_j(t_{i-1}) = 0, \quad p_j(t_i) = 1,
\]

and

\[
\hat{p}_k = v_k, \quad p_k(t_{i-1}) = 0, \quad p_k(t_i) = 1.
\]

The constraints at \( t_{i-1} \) and \( t_i \) will ensure that the reparameterization is braid-point feasible. For collision avoidance we will define a safety separation region, as depicted in Fig. 7, to be the region from the intersection point to the point where it is possible for the two agents to be within \( \delta_{jk} \) of each other, i.e., the set \( \left[ \frac{\Delta - \delta}{2\Delta}, \frac{\Delta + \delta}{2\Delta} \right] \) for \( \delta \in [0, \Delta] \) such that \( \left\| \gamma_i^j(p_j) - \gamma_i^k(p_k) \right\| \geq \delta_{jk} \) for all \( p_j, p_k \in [0, 1] \), where we let \( \delta \) be the distance from the intersection point to the boundary of the safety separation region along the path. Note that if the geometry of the braid strands are straight lines, the distance \( \delta \) can be easily computed by \( \delta = \delta_{jk} \csc(\theta) \), where \( \theta \) is the angle between the two intersecting lines, and this would also imply the parameters get as close as \( 2\delta_{jk} \csc(\theta/2) \). Similarly, when the geometry is “city-block”-like, we can find \( \delta = \delta_{jk} + \frac{h}{2(N-1)} \) and the agent get as close as \( 2\delta_{jk} \). For more general geometries, \( \delta \) may be conservatively selected. At \( \bar{t}_i := \frac{t_i - t_{i-1}}{2} \), i.e., half-time along the braid step, we will require one parameter to exit the safety separation region as the other one is about to enter it. All these cases are

will map the agents at braid step \( k \) from the set of braid points \( B_N^{k-1} \) to the next set of braid points \( B_N^k \) with at most \( m \) interactions total, and at most one interaction per agent.

If the geometric interpretation of the braid string is restricted to the case of only pairwise interactions at every braid step, then it is possible to devise a hybrid strategy with the desired properties, which we can then compose together to achieve the desired interaction patterns. As such, we will restrict the geometric interpretation of braid strings to those satisfying pairwise interactions as in Lemma 1.

**Restriction 1:** As a bijective map from one set of braid points to another, the braid \( \sigma(k) \) will only contain generators whose indices are two or more apart, i.e., for some \( h : \mathbb{N} \rightarrow \{0, \ldots, N - 1\} \)

\[
\sigma(k) = \sigma_{h(1)} \cdot \sigma_{h(2)} \cdot \cdots, \text{ where } |h(i) - h(j)| \geq 2 \forall i \neq j
\]

As an example, suppose that a braid string contains the substring of three concatenated generators \( \sigma_1 \cdot \sigma_3 \cdot \sigma_2 \). The first two generators, \( \sigma_1 \cdot \sigma_3 \), may take place simultaneously at braid step 1 without incurring in more than one interaction per agent with another agent, but we would require an additional braid step for \( \sigma_2 \) in order to avoid multiple interactions in the same step. This is illustrated geometrically in Fig. 6. In (a), an agent interacts with more than one other agent since all the generators are not at least two indices apart and take place at the same braid step. Note that in the remaining cases (b)-(d), we introduce intermediate braid points while retaining the desired level of interaction, final configuration of the agents, and reducing to pairwise interactions.

Braid strings that satisfy Restriction 1 will have at most interactions involving two agents at any given braid step. So
illustrated in Fig. 7. This way we guarantee that the two parameters are never inside the region simultaneously and thus their separation will always be of at least $\delta_{jk}$.

The optimality conditions for equation (1) are given by

\[
\lambda_j = \lambda_k = 0, \quad \dot{p}_j = -\lambda_j, \quad \dot{p}_k = -\lambda_k, \quad p_j(t_i) = \frac{\Delta + \delta}{2\Delta}, \quad p_k(t_i) = \frac{\Delta - \delta}{2\Delta}, \quad p_j(t_i) = p_k(t_i) = 1.
\]

Since $\lambda_j = \lambda_k = 0$, we will get that $\lambda_j$ and $\lambda_k$ are piecewise constant. By using the midway condition, we have that for $t \in (t_{i-1}, t_i)$

\[
p_j(t_i) = \int_{t_{i-1}}^{t_i} v_j \, dt = \frac{\Delta + \delta}{2\Delta}
\]

and similarly for $t \in (t_i, t_{i+1})$, the terminal condition tells us that

\[
p_j(t_i) = \int_{t_i}^{t_{i+1}} v_j \, dt = \frac{\Delta + \delta}{2\Delta}
\]

where the sign in the numerator is determined by the interpretation given to the braid strand going “under” (e.g., the agent crosses the intersection point first) or “over” (e.g., the agent crosses the intersection point second), and in cases where there are no intersections we set $\delta = 0$ in the numerator.

It is possible to come up with an upper bound on the length of the braid attainable under this mixing scheme. Under Restriction 1 the following theorem provides an upper bound on the mixing limit.

**Theorem 2:** Given the maximum safety separation $\bar{\delta}$ and bounds on the agents’ velocities such that $v_{min} \leq v_j(\bar{\delta}) \leq v_{max}$ for $N$-agent braids that can be performed in a space of height $h$ and length $\ell$ in time $T$ is bounded above by

\[
M^* \leq \min \left\{ \ell \sqrt{4h^2 - \bar{\delta}^2 (N - 1)^2} \frac{1}{\delta h}, \frac{(N - 1) (v_{max}T - (\ell + \bar{\delta})) - 1/2}{h} \right\}.
\]

**Proof:**
Consider $\sigma \in \Sigma^N$. Since at any step an agent can either interact with another agent or move straight ahead, the maximum mixing will be achieved if an agent interacts with another agent at every step. Thus, at every step it must be enforced that there are no collisions. Assuming a uniform partition of the time window, the arclength-normalized parameter velocity will be given by

\[
v_j = \begin{cases} \frac{(M \Delta \pm M \bar{\delta})}{T \Delta} & \text{if } t \in \left( \frac{1}{M} T, \frac{2}{M} T \right) \\ \frac{(M \Delta \pm M \bar{\delta})}{T \Delta} & \text{if } t \in \left( \frac{2}{2M} T, \frac{3}{2M} T \right) \end{cases}
\]

for $i = 1, 2, \ldots, M$, where the sign on the numerator depends on the interpretation of whether the “strand” goes over or under, $\Delta$ is the arclength of the strand geometry connecting two braid point, assumed equal at each braid step and for both agents due to symmetry, and $\bar{\delta}$ is the safety separation distance along the braid as described above. The total braid path length $\Delta$ heavily depends on the geometry of path. However, for the sake of obtaining bounds on the mixing limit, we may assume that the braid points are uniformly distributed in height and length, such that the length of sufficiently regular paths will be bounded by

\[
\sqrt{\left( \frac{h}{N - 1} \right)^2 + \left( \frac{\ell}{M} \right)^2} \leq \Delta \leq \frac{h}{N - 1} + \frac{\ell}{M} \tag{3}
\]

where the lower bound assumes straight lines connecting the braid points and the upper bound assumes “city block”-like paths stepping midway between the two points. But after normalizing the bounds on the parameter velocity, we see that

\[
0 \leq \frac{M (\Delta - \bar{\delta})}{T \Delta} \leq \frac{M (\Delta + \bar{\delta})}{T \Delta} \leq \frac{v_{max}}{\bar{\delta}} \tag{4}
\]

The lower bound of (4) tells us that for the parameter to not go backwards we need $\bar{\delta} \leq \Delta$. To ensure this, we set $\delta = \delta \csc \theta \leq \sqrt{\left( \frac{h}{N - 1} \right)^2 + \left( \frac{\ell}{M} \right)^2} \leq \Delta$, and since we require $\bar{\delta} \leq \frac{h}{N - 1}$ for collision-free braid points, using the geometric relationships to solve for $M$ yields

\[
M \leq \frac{\ell \sqrt{4h^2 - \bar{\delta}^2 (N - 1)^2}}{\Delta h}.
\]

Similarly, the right-hand side inequality of (4) tells us that $\Delta \leq \frac{v_{max}T - (\ell + \bar{\delta})}{M}$, and to ensure this we set $\Delta \leq \frac{h}{N - 1} + \frac{\ell}{M} \leq \frac{v_{max}T - (\ell + \bar{\delta})}{M}$. Solving for $M$ yields

\[
M \leq \frac{(N - 1) (v_{max}T - (\ell + \bar{\delta})) - 1/2}{h}.
\]
Thus
\[
M^* \leq \min \left\{ \frac{\ell \sqrt{4h^2 - \delta^2}(N - 1)^2}{\delta h}, \frac{(N - 1)(v_{\text{max}}T - (\ell + \delta))}{h} - \frac{1}{2} \right\}.
\]

Theorem 2 provides a compact expression to obtain an upper bound on the mixing limit that abstracts away strand geometry. It provides a notion of the whether or not desirable mixing levels are achievable in the space, regardless of what the actual movement patterns to achieve these mixing levels are (encoded in the braid string of length \(M \leq M^*\)). Fig. 8 includes a plot of the bound on the mixing limit for varying number of agents and time window size.

C. Non-Rectangular Regions

Up to this point, only the problem of braiding on a rectangular region of height \(h\) and length \(l\) has been considered. On this region, the braid points were uniformly distributed along both dimensions and bounds on the mixing limit were provided through the use of the proposed braid controller. In this section, the scheme is extended to more generally shaped regions, e.g., the road on Fig. 9. As has been done previously, discussion begins by first considering the two agent case.

Consider the two agents attempting to perform a braid of length one on the arbitrarily curved region on Fig. 9b. Connecting the braid points together results in the quadrilateral depicted in the red dotted line. Let this quadrilateral be considered as the space where the agent needs to perform a braid of length one, rather than the curved region itself. Note that as longer length braids are included in this road segment, better approximations of the curved region will be obtained by these composition of quadrilaterals. This is depicted in Fig. 9c.

The strategy for performing a mixing strategy in curved region will be to transform the curved region into a straightened rectangular region of known height and length, as illustrated in Fig. 10a. In this way the braid controller can be fashioned as in previous sections and the resulting braid controller can be transformed back into the actual curved region. After distributing the braid points on both the curved region and the rectangular region, as in Fig. 10a, the next step is to find a transformation to map between these two regions.

Let agents \(j\) and \(k\) interact in at braid step \(i\). Denote \(S_{i,j}^r\) to be the quadrilateral formed by connecting together the braid points \(\xi_i, j, \xi_{i-1}, j, \xi_{i,j}, \text{and} \xi_{i,k}\), like the one depicted in Fig. 9b. Let \(S_{i,j}^q\) be a rectangular plane of specified height and length whose corners are given by \(\xi_r(i - 1,j), \xi_r(i - 1, k), \xi_r(i, j), \text{and} \xi_r(i, k)\).

With knowledge of these braid points and through the use of a projective transform as in [30], it is possible to obtain a local diffeomorphism that maps from a rectangle of specified height and length to the convex arbitrarily shaped quadrilateral, i.e., \(T_{i,j} : S_{i,j}^r \rightarrow S_{i,j}^q\).

Note that by selecting transforms based on the corners of these quadrilaterals, a continuous curve that spans across the boundary between \(S_{i,j}^r\) and \(S_{i+1,j}^r\) might be mapped to a discontinuous curve that spans across the boundary between \(S_{i,j}^q\) and \(S_{i+1,j}^q\) when transformed using \(T_{i,j}\) and \(T_{i+1,j}\) in their respective spaces. However, there will certainly be continuity in the mapping of curves passing through the braid points, since these points are shared by the quadrilaterals and are used to compute the transforms, i.e., \(T_{i,j}(\xi_r(i, j)) = \xi(i, j) = T_{i+1,j}(\xi_r(i, j))\).

Recall the braid controller proposed for the rectangular
region. For agent \( j \), at braid step \( i \), this was given by

\[
v_j(t) = \begin{cases} 
\frac{1}{\Delta} \left( \frac{\Delta - \delta}{\tau_{i-1}} \right) & \text{if } t \in (t_{i-1}, t_i], \\
\frac{1}{\Delta} \left( \frac{\Delta + \delta}{\tau_{i-1}} \right) & \text{if } t \in [t_i, t_{i+1}).
\end{cases}
\]

where it is expected that \( T^{-1}_{i,j}(x(t_{i-1})) = \xi_r(i - 1, j) \) and \( T^{-1}_{i,j}(x(t_i)) = \xi_r(i, j) \). In the expression, \( \Delta \) corresponds to the length of the geometric path agent \( j \) must follow to move between \( \xi_r(i - 1, j) \) and \( \xi_r(i, j) \), while \( \delta \) corresponds to the distance along the path agent \( j \) switches velocities in order to avoid collisions. Note that for a given parameterization of the geometric path \( \gamma_{i,r}(p) \) in \( S^q_{i,j} \) with parameter \( p \in [0, 1] \), the arclength \( \Delta \) may be computed as follows

\[
\Delta = \int_0^1 \sqrt{\gamma_{i,r}^2(p) \gamma_{i,r}^2(p)} \, dp.
\]

If the geometric curve is directly given in \( S^q_{i,j} \) as \( \gamma_i^j(p) \) such that the curve in \( S^q_{i,j} \) may be parameterized as \( \gamma_{i,r}(p) = T^{-1}_{i,j}(\gamma_i^j(p)) \), then the arclength may be computed by

\[
\Delta = \int_0^1 \sqrt{\gamma_i^j(p)^\top M(\gamma_i^j(p)) \gamma_i^j(p)} \, dp.
\]

where \( M(\gamma_i^j(p)) = (DT_{i,j}^{-1}(\gamma_i^j(p)))^\top DT_{i,j}^{-1}(\gamma_i^j(p)) \) and \( DT_{i,j}^{-1} \) is the Jacobian of \( T_{i,j}^{-1} \). In the special case where the geometry is given by straight lines, then \( \Delta = \|\xi_r(i,j) - \xi_r(i - 1,j)\| \).

Finally, it is of interest to find \( \delta \) in order to avoid collisions. If the parameterization \( \gamma_i^j(p) \) of the curve in \( S^q_{i,j} \) is known, then the safety separation ball may be set in \( S^q_{i,j} \) as before and the safety separation distance may be computed as

\[
\delta = \int_a^b \sqrt{\gamma_i^j(p)^\top M(\gamma_i^j(p)) \gamma_i^j(p)} \, dp.
\]

where \([a, b] \subset [0, 1]\), and if the agent is meant to braid “over” (resp. “under”) then \( \gamma_i^j(a) \) corresponds to the point along the curve where the agent enters the safety separation region (resp. where the two curves intersect) and \( \gamma_i^j(b) \) corresponds to the point along the curve where the two curves intersect (resp. where the agent exits the safety separation region).

In the special case where the geometry is given by straight lines, then by setting

\[
\gamma_i^n(p) = (1 - p)\xi(i - 1, n) + p\xi(i, n), \quad p \in [0, 1], \quad n = j, k.
\]

as the path agent \( j \) and \( k \) must follow, the intersection point \( s \) may be found by setting \( s = \gamma_i^j(\pi_j) = \gamma_i^k(\pi_k) \) where

\[
\begin{bmatrix} \pi_j \\ \pi_k \end{bmatrix} = A^{-1}(\xi(i - 1, k) - \xi(i - 1, j))
\]

with \( A = [(\xi(i, j) - \xi(i - 1, j)), -\xi(i, k) + \xi(i - 1, k)] \).

Recall that the distance in \( S^q_{i,j} \) from the intersection point \( s \), for the special case of the geometry being straight lines, was given by \( \delta = \delta_{jk} \csc(\theta) \) with \( \theta \) being the angle between these two lines, i.e., \( \theta = \cos^{-1} \left( \hat{x}_n^\top \hat{x}_k \right) \) where \( \hat{x}_n \) is the unit vector pointing towards the next point, i.e.,

\[
\hat{x}_n = \frac{(\xi(i,n) - \xi(i - 1, n))}{\|\xi(i,n) - \xi(i - 1, n)\|}, \quad n = j, k.
\]

By setting \( \gamma(p) = (1 - p)s \pm p\delta \hat{x}_j \), where sign depends on whether the braid goes “over” or “under,” it is possible to determine \( \delta \) in the non-rectangular plane directly as

\[
\delta_r = \int_0^1 \sqrt{(\pm \delta \hat{x}_j - s)^\top M(\gamma(p))(\pm \delta \hat{x}_j - s)} \, dp.
\]

With this information, \( v_j \) can be found in \( S^q_{i,j} \) for interactions between agents \( j \) and \( k \). Note that a reparameterization of the path can now be set equivalent to the desired trajectory of agent \( j \) by setting \( \gamma_{i,d}^j(t) = \gamma_{i,r}^j(v_j(t)) \) (see below). Thus, the braid controller parameter velocity \( v_j^q(t) \) for agent \( j \) in \( S^q_{i,j} \) will be given by

\[
v_j^q(t) = \left\| DT_{i,j}(\gamma_{i,d}^j(t)) \gamma_{i,d}^j(t) \right\|.
\]

This strategy was implemented in simulation over the curved region illustrated in Fig. 11. In the figure, agents are performing the mixing strategy given to them by a braid of length 80 on the curved region (top) and simultaneously on the straightened rectangular region (bottom). The parameters used for this simulation were \( \delta_{jk} = 7.7 \text{ cm} \) and \( v_{max} = 1.5 \text{ m/s} \) \( \forall j, k \), and \( T = 30 \text{ s} \).

IV. IMPLEMENTING BRAIDS

Section 11 approached the problem of multi-robot mixing from an execution level, where given a specification it is possible to synthesize a braid controller that satisfies the specification. In this section we consider the implementation level, that is, we address how to find controllers that are implementable on actual robotic systems that follow the braid controllers found in the previous section. We then validate the framework by implementing a mixing strategy on a team of six robots.

A. Optimal Tracking Controller

We will now utilize the braid parameter velocity to find the braid controller for the agents. By integrating \( \gamma_i^j(t) \) we obtain the braid parameterization of the path \( \gamma_i^j \)

\[
p_j(t) = \begin{cases} 
\frac{t - t_{i-1}}{(t_i - t_{i-1})} \Delta \pm \delta & t \in (t_{i-1}, t_i], \\
\frac{t - t_i}{(t_i - t_{i-1})} \Delta \pm \delta + \frac{\Delta \pm \delta}{\Delta} & t \in [t_i, t_{i+1}).
\end{cases}
\]

where \( \Delta = 1 \text{ m/s} \) and \( \delta = 1 \text{ m} \). The control design will take place.
Assume that agents have single integrator dynamics, i.e., $\dot{x}_j = u_j$ with $y_j = x_j \in \mathbb{R}^2$. The agent’s controller will be found by optimally tracking the reparameterized path $\gamma_j^T(p_j(t))$ to minimize the cost

$$J(u_j) = \frac{1}{2} \int_{t_{i-1}}^{t_i} (x_j - \gamma_j^T) ^T Q (x_j - \gamma_j^T) + u_j^T R u_j \, dt \quad (6)$$

for $Q = Q^T > 0$ and $R = R^T > 0$, with constraints $\dot{x}_j = u_j$, $x_j(t_{i-1}) = \gamma_j^T(0)$, and $x_j(t_i) = \gamma_j^T(1)$. Using the standard variational argument together with Pontryagin’s minimum principle, the first order necessary conditions for optimality tell us that the optimal tracking controller $u_j^*$ is given by

$$u_j^* = -R^{-1}\lambda_j$$

where $\lambda$ is the so-called costate and satisfies

$$\dot{\lambda}_j = -Q (x_j - \gamma_j^T)$$

with unknown terminal condition $\lambda_j(t_i)$. Suppose that similarly to [31] Chapter 5.3 we can construct $\lambda_j$ as an affine combination of the unknown $\lambda_j(t_i)$ and the state, i.e.,

$$\lambda_j(t) = H(t)x_j(t) + K(t)\lambda_j(t_i) + E(t)$$

and similarly, the terminal state as

$$x_j(t_i) = \xi(i,j) = F(t)x_j(t) + G(t)\lambda_j(t_i) + D(t)$$

for some yet unknown functions $H, K, E, F, G,$ and $D$. One can differentiate these equations and manipulate the equations to obtain

$$\left(HR^{-1}H - \hat{H}\right) x_j + \left(HR^{-1}K - \hat{K}\right) \lambda_j(t_i) + \left(HR^{-1}E + Q\gamma_j^T - \hat{E}\right) = 0$$

and

$$\left(FR^{-1}H - \hat{F}\right) x_j + \left(FR^{-1}K - \hat{G}\right) \lambda_j(t_i) + \left(FR^{-1}E - \hat{D}\right) = 0.$$ 

In order to satisfy these equations for any value of $x_j(t)$ and $\lambda_j(t_i)$, the terminal conditions, and after noticing that $F = K^T$, we obtain that

$$\dot{H} = HR^{-1}H - Q, \quad H(t_i) = 0_{2 \times 2}$$

$$\dot{K} = HR^{-1}K, \quad K(t_i) = I_2$$

$$\dot{G} = K^T R^{-1}K, \quad G(t_i) = 0_{2 \times 2}$$

$$\dot{E} = HR^{-1}E + Q\gamma_j^T, \quad E(t_i) = 0_{2 \times 1}$$

$$\dot{D} = K^T R^{-1}E, \quad D(t_i) = 0_{2 \times 1}.$$ 

Note that $G(t_i) = 0$ and $\dot{G}(t) \geq 0$ for all $t$, which suggests that $G(t) \leq 0$ for $t < t_i$. If the problem is not abnormal, i.e., there exists a neighboring minimum solution, then $G$ will be invertible at some $t < t_i$. In particular, by solving backwards in time in the sequence $H \rightarrow K \rightarrow E \rightarrow G \rightarrow D$ up to $t = t_{i-1}$, we can find that

$$\lambda_j(t_i) = G^{-1}(t_{i-1}) (\xi(i,j) - K^T(t_{i-1})\xi(i-1,j) - D(t_{i-1}))$$

resulting in the feedback optimal trajectory tracking control law

$$u_j^*(x_j, t) = -R^{-1}\left[H(t)x_j(t) + K(t)G^{-1}(t_{i-1})\xi(i,j) - K^T(t_{i-1})\xi(i-1,j) - D(t_{i-1}) + E(t)\right]$$

where $H, K, E, G$ and $D$ are the solutions to terminal value problems in (7), which can be solved numerically backwards from $t_i$. As it turns out, these conditions are also sufficient for optimality as presented in the following theorem.

**Theorem 3:** The tracking controller in (8) is a minimizer to the cost functional (6) whose optimal value is given by

$$J(u^*) = \left[\xi(i-1,j)^T \left(\frac{1}{2}H - KG^{-1}K^T\right) \xi(i-1,j) + \xi(i-1,j)^T (KG^{-1}(\xi(i,j) - D) + E) + \varphi\right]_{t=t_{i-1}}$$

Fig. 11. Five agents performing a mixing strategy given by a braid of length 80. The top left plot represents the actual curved region the agents are mixing in and the top right is a close-up of the agents. The lower plot is the virtual “straightened” rectangular region where the design of the braid took place.
where $\varphi(t)$ is the solution to the terminal boundary problem
\[
\dot{\varphi} = \frac{1}{2} (\Lambda_j(t))^T R^{-1} \Lambda_j(t) - \frac{1}{2} \left( \gamma_j^i(t) \right)^T Q \gamma_j^i(t) \\
\varphi(t_i) = -\xi(i,j)^T \Lambda_j(t_i) \xi(i,j)
\]
with
\[
\Lambda_j(t) = E(t) + \\
K(t) G^{-1}(t_{i-1}) \xi(i,j) - K^T(t_{i-1}) \xi(i-1,j) - D(t_{i-1}).
\]

**Proof:**
As the control law was derived from the necessary conditions for optimality, we only need to show that it is sufficient for optimality. We will do so by leveraging the Hamilton-Jacobi-Bellman theorem [32, Chapter 2].

Note that the optimization problem is regular as there exists a $u_j$ that allows the Hamiltonian to achieve a minimum with respect to it, i.e.,
\[
\mathcal{H}(x_j, u_j, \lambda_j) \\
= \frac{1}{2} \left[ (x_j - \gamma_j^i)^T Q (x_j - \gamma_j^i) + u_j^T R u_j \right] + \lambda_j^T u_j \\
= \frac{1}{2} \left( u_j + \lambda_j R^{-1} \right)^T R \left( u_j + \lambda_j R^{-1} \right) - \frac{1}{2} \lambda_j^T R^{-1} \lambda_j \\
+ \frac{1}{2} \left( x_j - \gamma_j^i \right)^T Q \left( x_j - \gamma_j^i \right)
\]
which attains a minimum with respect to $u_j$ when
\[
u_j^* = -R^{-1} \lambda_j.
\]
Define $V(z,t)$ as

$$V(z,t) = \frac{1}{2} z^T H(t) z + z^T K(t) G^{-1}(t_{i-1}) \xi(i,j)$$

$$- z^T \left( K(t) G^{-1}(t_{i-1}) (K(t_{i-1}) \xi(i-1,j) - D(t_{i-1})) + E(t) \right) + \varphi(t)$$

where $\varphi(t)$ is as defined above. It can be verified that $V(z,t)$ satisfies the terminal condition $V(\xi(i,j),t_{i}) = 0$, that $\frac{\partial V(z,t)}{\partial z} \bigg|_{z=x_j} = \lambda_j$, and that is satisfies the Hamilton-Jacobi-Bellman equation, i.e.,

$$0 = \frac{\partial V(z,t)}{\partial t} \bigg|_{z=x_j} + \mathcal{H} \left( x_j, u_j^*(x_j,t), \frac{\partial V(z,t)}{\partial z} \bigg|_{z=x_j} \right).$$

As a consequence, $u_j^*$ is a minimizer to the cost functional and

$$J(u^*) = V(\xi(i-1,j),t_{i-1}).$$

As a final note, the terminal costate value $\lambda_j(t_{i})$ was computed using the initial conditions for the problem. However, as the gains involved are solved from terminal conditions, the choice of initial conditions is arbitrary, and evaluating at $t = t_{i-1}$ results in control law $u_j^*$ being open-loop in the terminal costate value. This could yield undesired results under the influence of disturbances and errors. To alleviate this, we can rewrite the terminal costate as a function of the current state value instead, i.e.,

$$\lambda_j(t_{i}) = G^{-1}(t_{i-1}) \left( \xi(i,j) - K^T(t_{i-1}) \xi(i-1,j) - D(t_{i-1}) \right) = G^{-1}(t) \left( \xi(i,j) - K^T(t) x_j(t) - D(t) \right)$$

in order to obtain the fully closed-loop optimal tracking controller

$$u_j^*(x_j,t) = -R^{-1} \left( \left[ (H(t) - K(t) G^{-1}(t) K^T(t)) x_j(t) + K(t) G^{-1}(t) (\xi(i,j) - D(t)) + E(t) \right) \right).$$

**B. Robotic Implementation**

In order to validate the above results in a practical setting, the braid controllers were implemented on a team of Khepera III differential-drive robots, which may be modelled using unicycle dynamics, i.e.,

$$\dot{x}_j = \left[ \nu_j \cos \theta_j \quad \nu_j \sin \theta_j \right]^T, \quad \dot{\theta}_j = \omega_j,$$

where $x_j \in \mathbb{R}^2$ is the robot’s position and $\theta_j$ its heading. The single integrator control $u_j$ from (9) can be mapped to unicycle dynamics as

$$\nu_j = \left[ \cos \theta_j \quad \sin \theta_j \right] \cdot u_j, \quad \omega_j = \left\{ \begin{array}{ll} \kappa \left[ -\sin \theta_j \cos \theta_j \right] \cdot \frac{u_j}{\|u_j\|} & \text{if } \|u_j\| > 1 \\ \kappa \left[ -\sin \theta_j \cos \theta_j \right] \cdot u_j & \text{otherwise} \end{array} \right.$$

for tuning gain $\kappa > 0$. Fig. [12](image) illustrates the team of 6 real robots executing the braid string

$$\sigma = \{\sigma_1, \sigma_3, \sigma_5\}, \sigma_2, \sigma_3, \sigma_4, \{\sigma_3, \sigma_5\}, \{\sigma_2, \sigma_4\}, \sigma_1$$

where the braids grouped in braces are executed simultaneously. A visual representation of the braid string geometry is being projected onto the robot workspace via an overhead projector mounted on the ceiling of the lab space. Fig. [13b](image) illustrates the minimum pairwise distance throughout the execution of the braid. It can be seen that the minimum distance achieved is approximately 0.132867m — greater than the size of the robots which is approximately 0.13m. Fig. [13a](image) illustrates the trajectories taken by the robots and how they follow the strand geometry, which were selected as straight lines, while still accommodating for the agents’ unicycle dynamics.
V. Conclusion

The notion of multi-robot mixing is presented as a framework with which to characterize rich movement patterns in a multi-robot system symbolically in terms of inter-robot interactions. Two controllers are presented which allow to the multi-robot system to achieve provably safe interactions, in the sense that agents get close without colliding, while meeting certain spatio-temporal constraints. Under these controllers, it is possible to obtain theoretical bounds on the mixing limit, the greatest level of mixing or inter-robot interactions achievable in a given region. Further, tracking controllers that optimally follow desired geometries are provided which can be mapped to classes of nonholonomic dynamical systems. The concepts presented here are validated on an actual multi-robot system executing a interaction pattern with a desired mixing level.

Not found in this paper is the question of how to select the interaction pattern that should be executed by these controllers. This question addresses the specification level of our symbolic approach. This discussion will be presented in a different paper where motion tasks are encoded in linear temporal logic, allowing for syntactically rich specifications at a high level, such as “agents 1 and 3 need to collect data before arriving at a goal location,” which are then translated into a mixing strategy encoded as a braid.

ACKNOWLEDGMENT

This work is partially supported under a grant from the Air Force Office of Scientific Research.

The authors would like to thank the members of the Georgia Robotics and Intelligent Systems Laboratory (GRITS Lab) for the many fruitful discussions.

REFERENCES

[1] T. Balch, “The impact of diversity on performance in multi-robot foraging,” in Proceedings of the third annual conference on Autonomous Agents. ACM, 1999, pp. 92–99.
[2] K. Sugawara, T. Kazama, and T. Watanabe, “Foraging behavior of interacting robots with virtual phenomone,” in Intelligent Robots and Systems, 2004. IROS 2004. Proceedings. 2004 IEEE/RSJ International Conference on, vol. 3, Sept 2004, pp. 3074–3079 vol.3.
[3] M. Bonert, L. Shu, and B. Benhabib, “Motion planning for multi-robot assembly systems,” International Journal of Computer Integrated Manufacturing, vol. 13, no. 4, pp. 301–310, 2000.
[4] W. Nguyen and J. K. Mills, “Multi-robot control for flexible fixtureless assembly of flexible sheet metal auto body parts,” in Robotics and Automation, 1996. Proceedings, 1996 IEEE International Conference on, vol. 3. IEEE, 1996, pp. 2340–2345.
[5] R. Groß, M. Bonani, F. Mondada, and M. Dorigo, “Autonomous self-assembly in swarm-bots,” Robotics, IEEE Transactions on, vol. 22, no. 6, pp. 1115–1130, 2006.
[6] E. Klavins, “Programmable self-assembly,” Control Systems, IEEE, vol. 27, no. 4, pp. 43–56, 2007.
[7] S. M. Pedersen, S. Fountas, H. Have, and B. Blackmore, “Agricultural robotsystem analysis and economic feasibility,” Precision agriculture, vol. 7, no. 4, pp. 295–308, 2006.
[8] N. Noguchi, J. Will, J. Reid, and Q. Zhang, “Development of a master–slave robot system for farm operations,” Computers and Electronics in agriculture, vol. 44, no. 1, pp. 1–19, 2004.
[9] R. C. Shah, S. Roy, S. Jain, and W. Brunette, “Data mules: Modeling and analysis of a three-tier architecture for sparse sensor networks,” Ad Hoc Networks, vol. 1, no. 2, pp. 215–233, 2003.
[10] I. Vasilescu, K. Kotay, D. Rus, M. Dunbabin, and P. Corke, “Data collection, storage, and retrieval with an underwater sensor network,” in Proceedings of the 3rd international conference on Embedded networked sensor systems. ACM, 2005, pp. 154–165.
[11] W. Zhao, M. Ammar, and E. Zegura, “Controlling the mobility of multiple data transport ferries in a delay-tolerant network,” in INFOCOM 2005. 24th annual joint conference of the IEEE computer and communications societies. Proceedings IEEE, vol. 2. IEEE, 2005, pp. 1407–1418.
[12] A. Howard, “Multi-robot simultaneous localization and mapping using particle filters,” The International Journal of Robotics Research, vol. 25, no. 12, pp. 1243–1256, 2006.
[13] N. E. ¨Ozkucur and H. L. Akin, “Cooperative multi-robot map merging using fast-slam,” in RoboCup 2009: Robot Soccer World Cup XIII. Springer, 2010, pp. 449–460.
[14] R. M. Murray, “Recent research in cooperative control of multivehicle systems,” Journal of Dynamic Systems, Measurement, and Control, vol. 129, no. 5, pp. 571–583, 2007.
[15] X. C. Ding, A. R. Rahmati, and M. Egerstedt, “Multi-uvu convoy protection: an optimal approach to path planning and coordination,” Robotics, IEEE Transactions on, vol. 26, no. 2, pp. 256–268, 2010.
[16] J. Hall, “Efficient spiral search in bounded spaces,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 4, no. 2, pp. 208–215, March 1982.
[17] W. Huang, “Optimal line-sweep-based decompositions for coverage algorithms,” in IEEE International Conference on Robotics and Automation, 2001.
[18] R. Hashemi, L. Jin, G. Anderson, E. Wilson, and M. Clark, “A comparison of searchpatterns for cooperative robots operating in remote environment,” in International Conference on Information Technology: Coding and Computing, 2001.
[19] J. Kim, F. Zhang, and M. Egerstedt, “Simultaneous cooperative exploration and networking based on Voronoi diagrams,” in IFAC Workshop on Networked Systems, October 2009.
[20] B. Tovar, L. Freda, and S. M. LaValle, “Using a robot to learn geometric information from permutations of landmarks,” in Contemporary Mathematics. American Mathematical Society, 2007, vol. 438, pp. 33–45.
[21] E. Artin, “Theory of braids,” Annals of Mathematics, vol. 48, no. 1, pp. 101–126, 1947.
[22] J. S. Birman, Braids, links, and mapping class groups. Princeton University Press, 1975, no. 82.
[23] C. Belta, A. Bicchi, M. Egerstedt, E. Frazzoli, E. Klavins, and G. J. Pappas, “Symbolic planning and control of robot motion,” IEEE Robotics and Automation Magazine, vol. 14, no. 1, pp. 61–70, March 2007.
[24] M. Egerstedt, X. Hu, and A. Stotsky, “Control of mobile platforms using a virtual vehicle approach,” Automatic Control, IEEE Transactions on, vol. 46, no. 11, pp. 1777–1782, 2001.
[25] T. Lozano-Perez, “Spatial planning: A configuration space approach,” Computers, IEEE Transactions on, vol. C-32, no. 2, pp. 108–120, Feb 1983.
[26] R. Ghrist, “Configuration spaces and braid groups on graphs in robotics,” AMS IP STUDIES IN ADVANCED MATHEMATICS, vol. 24, pp. 29–40, 2001.
[27] V. Kurlin et al., “Computing braid groups of graphs with applications to robot motion planning,” Homology, Homotopy and Applications, vol. 14, no. 1, pp. 159–180, 2012.
[28] Y. Diaz-Mercado and M. Egerstedt, “Multi-robot mixing using braids,” in Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on, Dec 2013, pp. 2001–2005.
[29] ——, “Multi-robot mixing of nonholonomic mobile robots,” in Control Applications (CCA), 2014 IEEE Conference on, Oct 2014, pp. 524–529.
[30] A. Criminisi, I. Reid, and A. Zisserman, “A plane measuring device,” Image and Vision Computing, vol. 17, no. 8, pp. 625–634, 1999.
[31] A. E. Bryson, Applied optimal control: optimization, estimation and control. CRC Press, 1975.
[32] A. Locatelli, Optimal control: An introduction. Springer Science & Business Media, 2001.