Examples

Synthesis of Collision Avoidance Protocol

Problem statement

In this scenario we have n UAVs, m altitude layers and q locations of interest. A UAV can ascend or descend to the altitude layer above or below its current altitude layer. A location is a predefined point on an altitude layer. Our aim is to automatically synthesize a control protocol that guarantees that each UAV is able to visit, infinitely often, all of the locations of interest. Additionally, there must be no collision between UAVs.

Approach

Our approach to this problem is centered around the idea of UAVs' operating regions. An operating region for a UAV is a polygon on the UAV's current altitude layer. We assume That the UAV will only fly within its operating region. With this assumption, we can guarantee that no collision will happen if the UAV with intersected operating region remain still until the intersection is resolved.

At each turn, each UAV will signal an intention to visit a specific location or to remain still. We assume that there cannot be two UAVs intending to fly to the same location to prevent deadlocks. We use this intention signal to update the operating regions for all the UAVs so that the intended locations are inside the operating regions of the UAVs. The algorithm for updating the operating regions ensures that the updated regions for all the UAVs do not intersect. Currently, there is no guarantee that the algorithm will terminate. The algorithm includes reshaping the operating regions and moving the UAVs to nearby location to resolve operating regions intersections. The UAV will then be given the command to fly to location it intended to fly to. It is assumed that the UAV will reach the location in one time step.

Problem setup

We model this problem by assigning three output variables, $l_i \in L$ for the layer that UAV_i should ascend or descend to, where L is the set of indices of the altitude layers, $t_i \in P$ for signaling UAV_i's intended location to fly to, where P is the set of indices of locations and $p_i, t_i \in P$ for the location that UAV_i should fly to. In addition, we assign $(n-1)^2$ input variables $c_{ij} \in B$, $i \neq j$ for indicating whether the operating region of UAV_i intersects with the operating region of UAV_j, where B is the boolean set and n is the number of UAVs and $i, j \in U$, which is the set of indices for the UAVs.

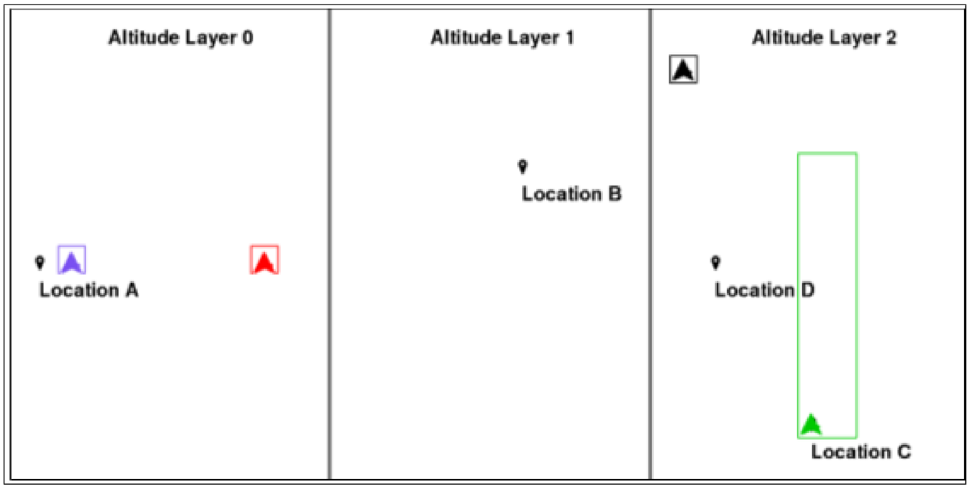

Figure 1: The airspace is divide into three equal layers numbered 0 to 2. Each of the four UAVs has a polygon surrounding it with the same color which defines the borders of the operating region of the UAV. The pinpoints point at the exact position of the corresponding location.

System specifications

We express the system's specifications in four sets of LTL specifications. First, UAV_i can only go to the locations that are in the same altitude layer ($\bigwedge_{i \in U} \square(l_i = u \implies \bigvee_{j \in C_v} p_i = v_j \vee p_i = \text{stay})$) where C_v is the set of indices of the locations defined on the altitude layer u. Second, UAV_i and UAV_j must not be at the same altitude layer if c_{ij} is true (their operating regions intersect!) ($\bigwedge_{i \in U} \bigwedge_{j \in U} \square(c_{ij} \implies l_i \neq l_j)$). Third, If UAV_i is given a command to go to x then it must have signaled its intent to do so at previous step ($\bigwedge_{i \in U} \square(p_i = x \implies t_i = x)$). Lastly, UAV_i will be repeatedly sent the command to fly over x ($\bigwedge_{i \in U} \square\lozenge(p_i = x)$).

Environment assumptions

We assume that if two UAVs have intersecting operating regions (c_{ij} is true) and one of the UAVs signaled an intention to go to a location then the intersection will be resolved (c_{ij} is false).

Synthesis of VIP Escort Protocol

Problem statement

In this scenario we have one main UAV we call "VIP", multiple support UAVs we call "escorts", one adversary UAV we call "enemy", and multiple pre-defined locations on the map. These locations are shown in green in fig 2. Our aim is to automatically synthesize a control protocol that guarantees the following four properties. First, The VIP can only fly from one location to another if it is being followed by one of the escorts. Second, The VIP cannot visit certain locations until at least one of the escorts have previously inspected by flying over the location. In order for the UAVs not to pass through the prohibited regions shown in red in fig 2, the UAVs must follow certain paths between the locations in green. For example, to go from the bottom right location to the upper right, the UAVs must pass through the location in the center in order to not fly over the prohibited regions.

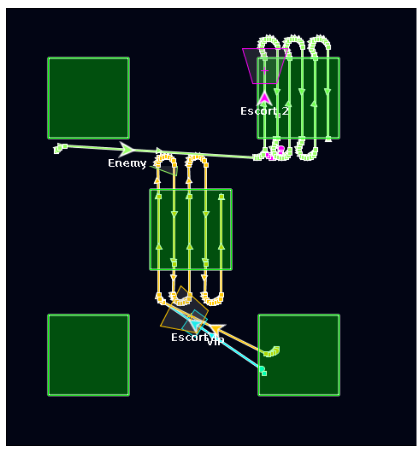

Figure 2: The escort UAV (blue) is tracking the VIP (yellow), allowing the VIP to fly from the bottom right location to the top right via the middle location.

Problem setup

We model this problem by assigning two output variables for each escort UAV. $g_i \in P$ is an output variable that indicates the location that UAV$_i$ should fly to where P is the set of location indices and $t_i \in B$, the second output variable, is an indicator for whether the escort UAV$_i$ should track the VIP or not, where B is the boolean set. As for the VIP, we assign one output variable $v \in P$ for the location that the VIP should fly to. We also assign one input variable to model the current location of the enemy $o \in P$. In addition, we assign the input variables $r_i \in B$ that indicate if the location was previously inspected.

System specifications

The system specifications for synthesizing the controller consist of five sets of LTL specifications. First, the escort must be at the same location as the VIP to be able to begin tracking it $\square(\bigcirc t_i \implies v = g_i)$. Secondly, The VIP cannot fly from a location to another without being followed by another UAV $\square(\bigcirc v \neq v \implies \bigvee_{i \in U} t_i)$. The third specification is that the VIP must always eventually visits a set of locations: $\bigwedge_{p \in P} \square \Diamond v = p$. the fourth is that a set of locations must always eventually be inspected $\bigwedge_{i \in P} \square \Diamond r_i$. Lastly, the VIP must not go to a location where the enemy is currently located $\square(v \neq o)$

Environment assumptions

We express the environment's assumptions in three sets of LTL specifications. First, the location is considered inspected when an escort visits the location $\square(g_i \implies r_i)$. Second, an inspected location remains inspected $\square(r_i \implies \bigcirc r_i)$ (it is also possible to set an expiration timer for the location to switch back to not inspected). The last assumption is modeling our assumptions about the enemy UAV; there are several way to model this, one simple way is to specify the locations that the enemy UAV does not go to: $\bigwedge_{d \in D} \square(o \neq d)$ where $D \subset P$ is the set of locations that the enemy does not go to.