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 1) each UAV is able to visit, infinitely often, all of the locations of interest and 2) there must be no collision between UAVs.

Approach

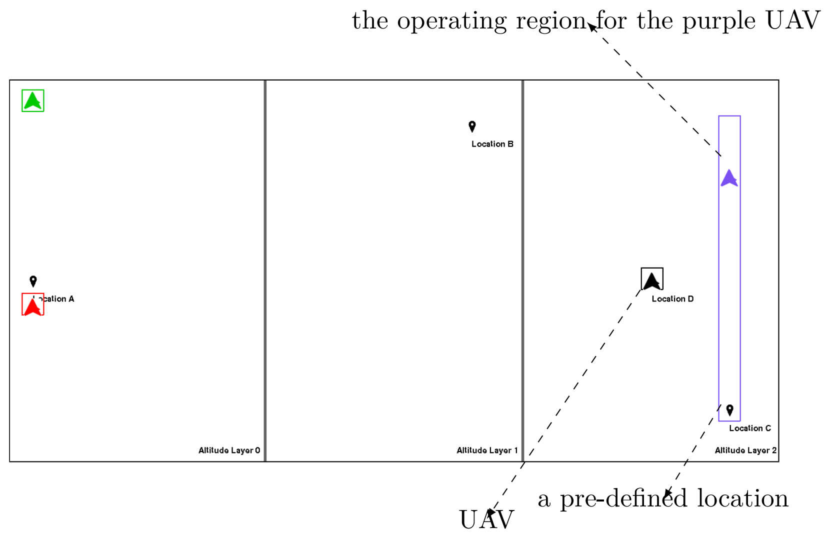

Figure 1:

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 $(n-1)$ input variables $c_{ij} \in B$ where B is the boolean set and $c_{ij} = c_{ji}$ to each UAV$_i$ where $i, j \in U$, which is the set of indices for the UAVs.

System specifications

We express the system's specifications in 4 sets of LTL specifications. First, UAV$_i$ can only go to the locations that are in the same altitude layer $(\bigwedge_{i\in U} \Box(l_i = x \implies p_i = x \bigvee_{i\in P} p_i = stay))$. Second, UAV$_i$ and UAV$_j$ must not be at the same altitude layer if c_{ij} is true (their operating regions intersect) $(\bigwedge_{i\in U}\bigwedge_{j\in U} \Box(c_{ij} \implies l_i \neq l_j))$. Third, If UAV$_i$ is given a command to go to x then it must have signaled its intent at the previous step $(\bigwedge_{i\in U} \Box(p_i = x \implies t_i = x))$. Lastly, UAV$_i$ will eventually be sent the command to go to x $(\bigwedge_{i\in U} \Box\Diamond(p_i = x))$.

Environment assumptions

We assume that if two UAVs have intersecting operating regions and one of the UAVs signaled an intention to go to a location then the intersection will be resolved: $(\Box\Diamond \bigwedge_{i\in U}\bigwedge_{j\in U}(\neg c_{ij}))$.

Synthesis of VIP Escort Protocol

Problem statement

In this scenario we have one main UAV we call "VIP", multiple "escort" UAVs, 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 three properties. 1) The VIP must not move from one location to another without being followed by one of the escorts. 2) The VIP cannot visit certain locations until at least one of the escorts have inspected the location within the last x steps. A location is considered inspected if an escort flies over the location. 3) 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 middle location so that they do not fly over the prohibited regions.

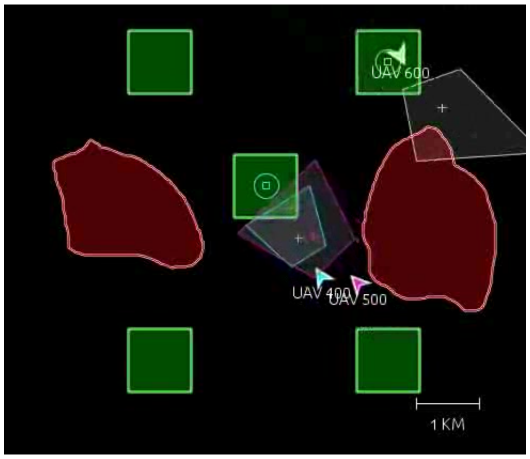

Figure 2:

Problem setup

We model this problem by assigning two output variables $g_i \in P$ for the location that UAV$_i$ should fly to where P is the set of location indices and $t_i \in B$ is an indicator for whether the escort UAV$_i$ should track the VIP or not, where B is the boolean set and one input variable $s_i \in P$ for the current location of UAV$_i$ where $i \in U$, the set of escort UAV indices. As for the VIP we assign one input variable $p \in P$ for the VIP's current location and one output variable $q \in P$ for the location that the VIP should fly to. In addition, we assign a variable $r_i \in B$ to indicate whether the location was previously inspected.

System specifications

The system specifications for synthesizing the controller consist of four sets of LTL specifications.1) The escort must be at the same location as the VIP to be able to follow it $\square(t_i \implies p = s_i)$.2) The VIP cannot fly from a location to another without being followed by another UAV $\Box(\bigwedge_{i\in U} \neg ti_i \implies \bigcirc p = p)$.3) The VIP must always eventually visits a set of locations: $\bigwedge_{p\in P} \Box\Diamond q$.4) a set of locations must eventually be inspected $\bigwedge_{i\in P} \Box\Diamond r_i$.

Environment assumptions

We make three sets assumptions on the environment. First, the location is considered inspected when an escort visits the location, $\Box(s_i \implies r_i)$. Second, an inspected location remains inspected. It is also possible to set an expiration timer for the location to switch back not inspected: $\Box(r_i \implies \bigcirc r_i)$. The last assumption is that all UAVs will reach their destinations they were commanded to fly to by the next time step: $\Box(g_i \implies \bigcirc s_i)$.