



En détails

Partitionnement du polygone

Nous cherchons à diviser l'aire **A d'un polygone en n partitions** d'aire $\frac{A}{n}$ (où n représente le nombre de drones). Nous effectuons ensuite une **triangularisation du polygone**. Après avoir choisi un vertex **V** de départ, nous choisissons un des deux triangles adjacents T_1 ou T_3 , d a n s notre exemple: T_1 .

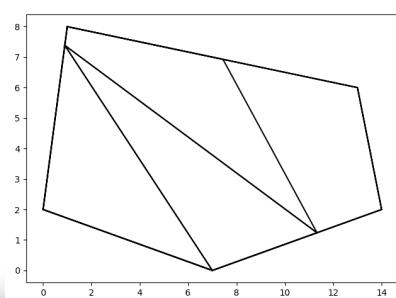
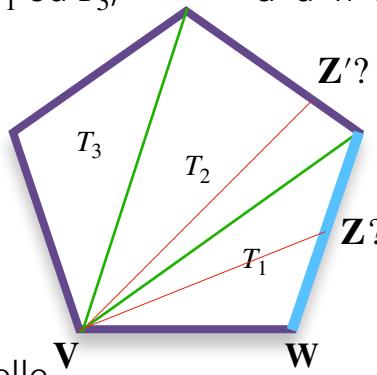
- Si celui-ci possède une **aire égale** à celle recherchée, nous avons une première sous-région.
- Si l'aire recherchée est **plus petite** de celle de T_1 il faut alors trouver les coordonnées (x, y) d'un **point Z** inconnu situé sur le **côté -du triangle- opposé à V**, tel que l'aire du triangle VWZ correspond à celle recherchée.

Pour cela nous avons construit une formule basée sur la **formule de Shoelace**. Cette dernière est utilisée pour trouver l'aire d'un polygone en connaissant les coordonnées de tous ses vertex. Par exemple dans le cas d'un triangle nous avons:

$$A = \left| \frac{1}{2} (x_1 \cdot (y_2 - y_3) + x_2 \cdot (y_3 - y_1) + x_3 \cdot (y_1 - y_2)) \right|$$

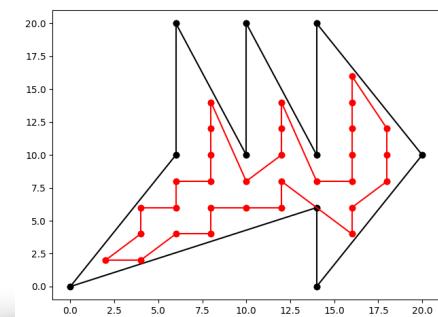
- Cependant, si $\frac{A}{n} > \text{aire}(T_1)$ **nous répétons les deux étapes** précédentes dans le triangle adjacent. Dans l'exemple ci-dessus nous chercherons donc un point Z' faisant partie du triangle T_2 .

Toutes ces opérations sont répétées **jusqu'à obtenir n partitions exactes** comme dans l'exemple ci-contre.



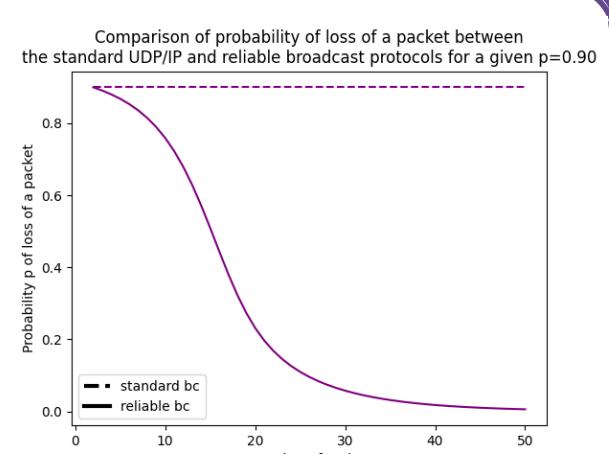
Résolution du chemin optimal

Nous construisons à l'intérieur de la sous-région une **grille imaginaire de points** telle qu'un carré correspond au **champ visuel de chaque drone**. Le drone doit trouver un cycle qui passe par tous les points de la grille une et une seule fois (**cycle Hamiltonien**), tout en garantissant le **chemin le plus court**. Nous avons donc réduit notre problème à une instance du **problème du Voyageur de Commerce** qui recherche le chemin le plus court entre des villes en revenant finalement à la ville de départ. Ici les **villes** sont **symbolisées par les points** de la grille -en pratiques des noeuds d'un **graphe complet**. Grâce aux outils OR-Tools de **Google**, la solution est calculée très rapidement afin de garantir dynamisme et **réactivité**. De plus, le fait d'avoir beaucoup de points dans la grille garantit une **couverture visuelle totale** de la sous-région.



Protocole de communication et résultats

Plusieurs types de serveurs tournent sur chaque drone, chacun ayant un rôle spécifique dans la diffusion d'informations. Le premier s'occupe de la communication avec le **monde extérieur**, et le deuxième sera responsable de la **communication interne** du réseau. À travers cette structure, l'échange des paquets (étant propres à chaque serveur) permet de transmettre diverses informations entre les différents drones. Comme par exemple, la **disponibilité des drones** sur le réseau, l'envoi d'images sur ce que chaque **drone observe** à sa position courante, mais également le rassemblement des différentes données, afin de **monitorer l'avancée** depuis le monde extérieur. L'efficacité de celui-ci suit une loi binomiale $X \sim B(n, p)$ où p est la probabilité de perte d'un paquet. La probabilité de perdre k paquets est donc de : $\psi(n) = p [(1 - \psi(n - 1))p + \psi(n - 1)]^{n-2}$ t.q. $n - 2 \geq 0$. Grâce au « Reliable Broadcast », les taux de perte sont très faibles, comme nous le constatons dans le schéma ci-dessus.



Conclusions

En conclusion, les aspects telles que la décentralisation, l'autonomie et le dynamisme confèrent à notre système l'habileté d'agir efficacement dans des missions de repérages à haut risques.