Unité 6 : utiliser les bibliothèques TI Hub & TI Rover

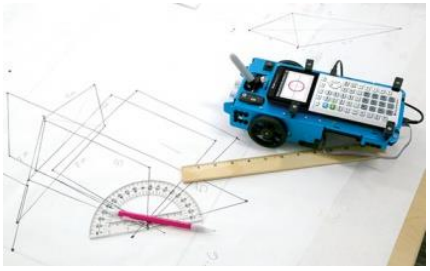
Application : Représenter un parcours

Dans cette application de l'unité 6, vous allez connecter le TI-Innovator™ Rover à l'aide de la bibliothèque **TI Hub** et construire un script permettant d'enregistrer des coordonnées de points lors d'un parcours, puis de les représenter graphiquement.

Objectifs :

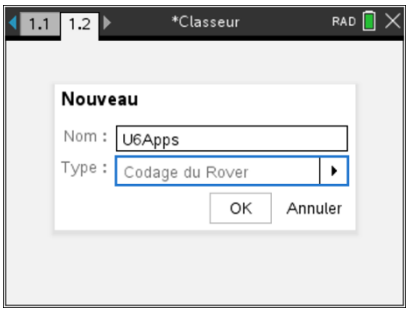
- Découvrir le module **TI Rover.**
- Écrire et utiliser un script permettant d'utiliser le TI-Innovator™ Rover et ses actionneurs associés.
- Utiliser une boucle fermée.
- Représenter graphiquement des données.

Vous allez, dans cette leçon, réaliser un script donnant au TI-Innovator™ Rover la possibilité d'effectuer un parcours correspondant au dessin d'un polygone.
Les coordonnées des sommets seront sauvegardées dans des listes puis représentées graphiquement à l'aide des instructions de la bibliothèque **TI Plot**.

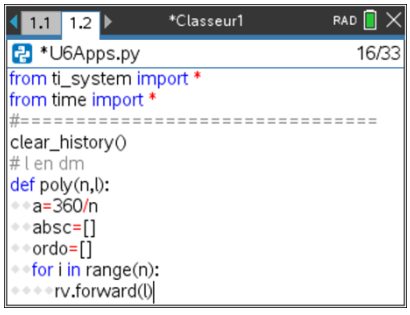


Mise en œuvre :

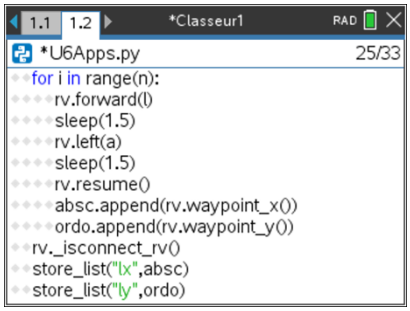
- Commencer un nouveau script et le nommer U6APPS.
- Choisir un script de type « Codage du Rover ».
- Valider en appuyant sur la touche enter.
- Importer également la bibliothèque **TI Plotlib**



- Bien que cela ne soit pas impérativement nécessaire, les instructions permettant d'effacer l'écran et de ne pas afficher le curseur sont toujours d'un rendu plus agréable. Insérer l'instruction **clear_history()** se trouvant dans la bibliothèque **TI System**.
- Créer une fonction **poly(),** prenant en argument $\boldsymbol{n}$ le nombre de côtés du polygone et $\boldsymbol{l}$ la longueur en unité par défaut pour le TI-Innovator™ Rover, c'est-à-dire le dm.
- L'angle au sommet de chaque polygone sera donc égal $a = \frac{360}{n}$.
- Créer deux listes vides **absc** et **ordo**, destinées à recevoir les coordonnées de chacun de ces sommets.



- Créer une boucle ouverte de taille n.

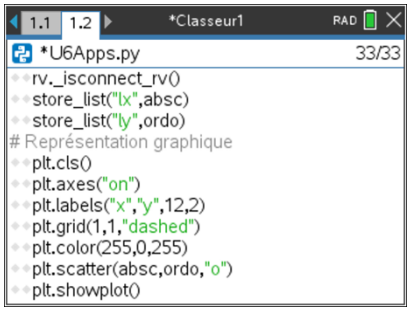


Les instructions suivantes se trouvent dans la bibliothèque **TI Rover.**

- Faire avancer le rover de $\boldsymbol{l}$ décimètres pour $\boldsymbol{n}$ fois.
- Tourner sur la gauche d'un angle $\boldsymbol{a}$**.**
- Mettre entre chaque étape un délai de 1,5s.
- Stocker dans les listes x et y les coordonnées des points.
- Déconnecter le Rover à la fin du parcours.
- Exporter les coordonnées **absc** et **ordo** respectivement dans les listes **lx** et **ly**, pour éventuellement faire une représentation graphique des données dans une application de la TI-Nspire™ (Graphiques, Tableur&Listes…).

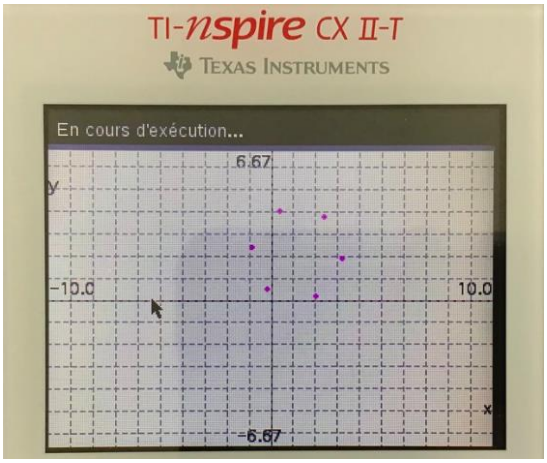
Conseil à l'enseignant : les instructions **rv.waypoint_x( )** et **rv.waypoint_y( )** se trouvent dans le menu **5 Path** de la bibliothèque **TI Rover**.

- Passer ensuite à la représentation graphique ; celle-ci est ici incluse dans la fonction, mais il est toujours possible de créer une fonction graphe séparément comme nous avons pu le faire dans les leçons précédentes (Unité 5).



Fonctionnement du script :

Exécuter votre script et appeler la fonction **poly( )** en lui fournissant par exemple les arguments poly(4, 1), afin de dessiner dans un premier temps un carré de 1 dm de côté.



Poursuivre par un essai avec un hexagone de 2 dm ce qui fournit la représentation suivante.

Conseil à l'enseignant : Pour ce type d'exercice, éviter d'utiliser les instructions **rv.pathlist_x( )** et **rv.pathlist_y( )**.
En fait lors du tracé d'un segment, la calculatrice enregistre les coordonnées des points d'un segment du polygone, puis réenregistre les coordonnées du dernier point comme premier point du segment suivant. D'autant plus que nous avons placé, entre chaque tracé, une temporisation de 1s.
Ainsi les instruction rv.path… sont dans notre cas inappropriées.
En utilisant les instructions **rv.pathlist_x( )** et **rv.pathlist_y( ),** on obtiendrait deux fois les coordonnées des extrémités.

Remarque : Ajuster le format de votre grille, en fonction des polygones que vous souhaitez tracer. Celle-ci a volontairement été réglée ici aux paramètres par défaut, afin d'observer la précision du tracé du TI-Innovator™ Rover.

Prendre garde également à la nature de la surface sur laquelle se déplace le robot. Celle-ci ne doit pas opposer trop de résistance au déplacement ou, au contraire, favoriser les glissements.