LSL

Véhicule Automatique

par Bestmomo Lagan

Version beta 1.03

19 juin 2009

Table des Matières

Introduction....4
La problématique....4
Les solutions classiques....4
La solution proposée....4
Prise en main....6
Contenu du pack....6
Mise en place de la trajectoire....6
Détection de la trajectoire....13
Paramétrage du mobile....16
La Trajectoire....23
La spline cubique d'Hermite....23
Vocabulaire....24
Trajectoire (Path)....24
Trajectoire fixée....24
Segment....24
Balise....24
Borne....24
Mobile....24
Detector....24
Code de couleur des balises....24
Balise verte....24
Balise jaune....25
Balise rouge, grise ou bleue....25
Principe de réglage d'un segment....25
Types de trajectoire et repérage....26
Trajectoire ouverte....26
Trajectoire fermée....27
Mise en place de la trajectoire....28
Première balise type F (First)....28
Premier segment....28
Segments intermédiaires....30
Verrouillage d'une balise....30
Fin et fixation d'une trajectoire....31
Fin d'une trajectoire ouverte....31
Fin d'une trajectoire fermée....31
Modification d'une trajectoire fixée....31
Le mobile....32
Lecture et mémorisation de la trajectoire par le detector....32
Déplacement sur la trajectoire....33

Le script du mobile....33
Gestion manuelle du déplacement (mode manuel)....33
Gestion automatisée du déplacement (mode auto)....34
Mode auto simple....35
Commandes d'arrêt et d'affichage de texte à l'arrêt....35
Commandes intermédiaires....36
Exemple de script en mode auto simple....36
Mode auto avec arrêt permanent....37
Exemples de scripts en mode auto avec arrêt permanent....37
Mode auto avec stations....38
Exemples de scripts en mode auto avec STATION....39
Contraintes syntaxiques en mode auto avec STATION....43

Introduction

Déplacer un véhicule dans SL est relativement facile en utilisant les fonctionnalités « Véhicule » intégrées à LSL. Cela permet d'obtenir des véhicules paramétrables selon les besoins (avion, moto, sous-marin…) que l'on peut piloter à partir du clavier. C'est par contre une toute autre histoire si on veut créer un véhicule autonome à mouvement programmé ou semi-automatique.

La problématique

Il est relativement facile de déplacer un objet en ligne droite dans SL. Pour obtenir un mouvement fluide on utilise la fonction **llMove2Target**. L'objet doit être du type « physique », ce qui limite son nombre de primitives à 32. Pour adapter la rotation on utilise la fonction **llRotLookAt**. On peut ainsi déplacer un objet de point en point en ligne droite.

Pour obtenir un déplacement le long d'une courbe il est possible de diviser celle-ci en petits segments de droite. Si ces derniers sont suffisamment courts on peut assimiler le déplacement à celui qui suit la courbe. Si dans le principe cette approche est satisfaisante elle révèle rapidement ses difficultés techniques. Comment définir les points de référence ? Comment le véhicule peut-il les identifier ? Comment gérer ces petits déplacements ?

Les solutions classiques

Différentes approches plus ou moins astucieuses ont été proposées. La plus lourde consiste à créer un système de balisage repéré par le véhicule. Il faut alors prévoir la mise en place de ces balises et la communication entre le véhicule et les balises (sensor, message…). La plus légère consiste à programmer le véhicule, par exemple en le positionnant aux différents points de passage et en lui faisant mémoriser ceux-ci. On voit facilement le caractère laborieux de cette approche. Aucune des démarches proposées n'apporte de solution élégante à la problématique évoquée.

La solution proposée

La solution proposée se situe un peu au carrefour des deux grands axes évoqués ci-dessus, en associant un système de balisage passif performant et une gestion de véhicule autonome.

Dans le principe général on commence par mettre en place les balises avec un système de génération simple et une visualisation de la trajectoire au moyen de bornes déposées le long du parcours. On peut ajuster avec précision la forme des courbes en se fondant sur une spline cubique d'Hermite (voir chapitre suivant). A l'issue de cette mise en place les balises deviennent totalement passives (les valeurs sont mémorisées dans la description de la

primitive) pour ne nécessiter aucune ressource au niveau du serveur. Les balises sont chaînées entre elles par leur clé.

Un objet, le detector, situé à proximité d'une balise, lit les valeurs de celle-ci puis des suivantes grâce au chaînage réalisé entre les balises. Lorsqu'il a mémorisé l'ensemble des valeurs de la trajectoire il les liste dans le Chat. Ces données sont ensuite stockées dans un notecard et servent de référence pour le mobile. Les balises ne servent alors plus à rien. Elles ne sont donc nécessaires que pendant la phase d'initialisation du véhicule.

Prise en main

Contenu du pack

Le pack contient 3 objets :

- balise
- dectector
- HUD mobile

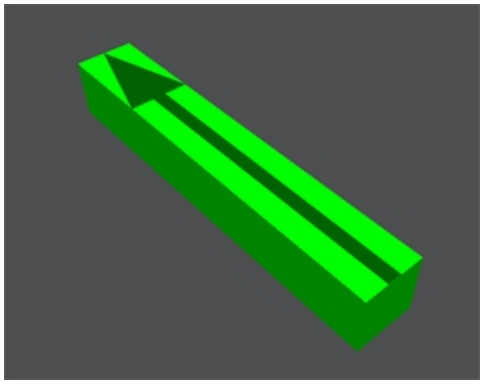
Il contient également 4 scripts :

- mobile
- sequencer
- syntaxor
- station

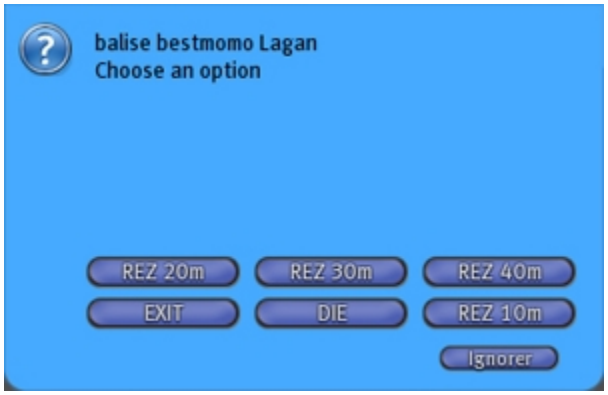
Créez un répertoire dans votre inventaire pour ranger tout ça !

Mise en place de la trajectoire

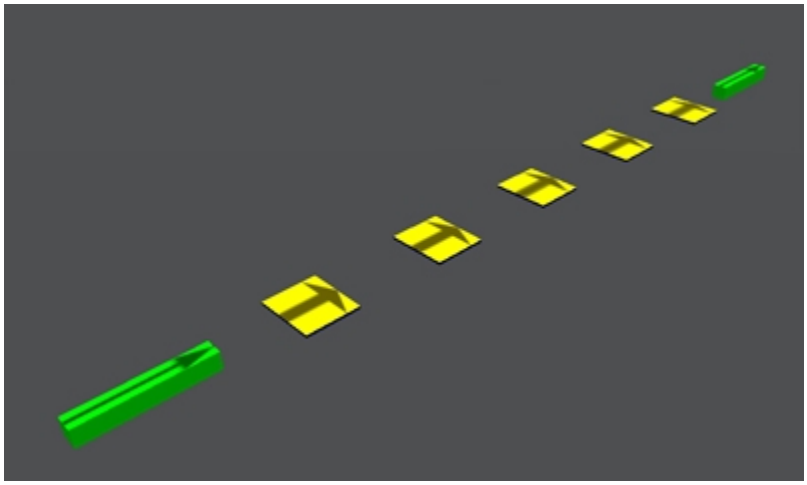
Posez un objet « balise » sur l'emplacement où vous désirez créer la trajectoire :



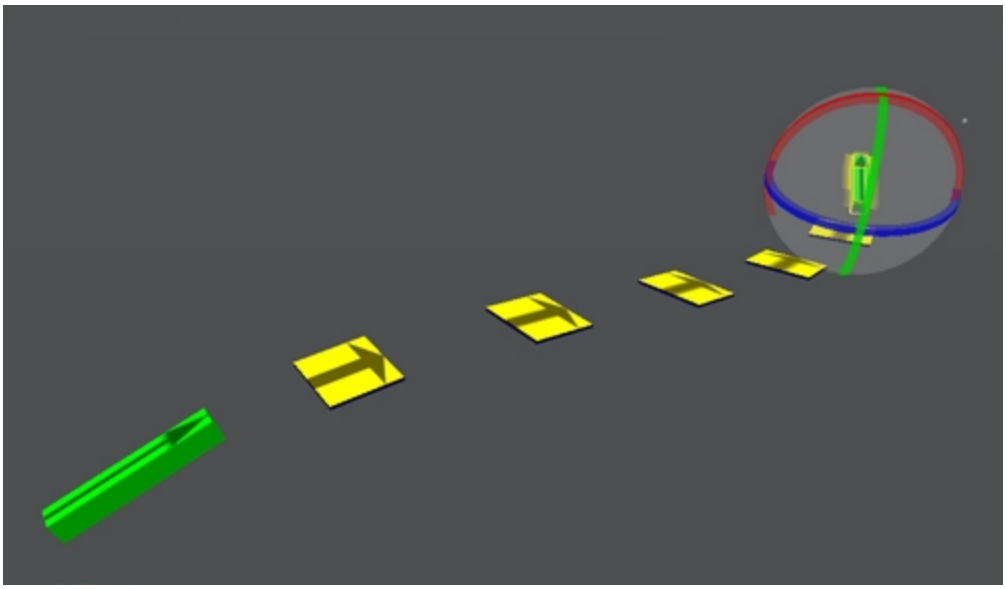
Cliquez sur la balise pour obtenir le menu :



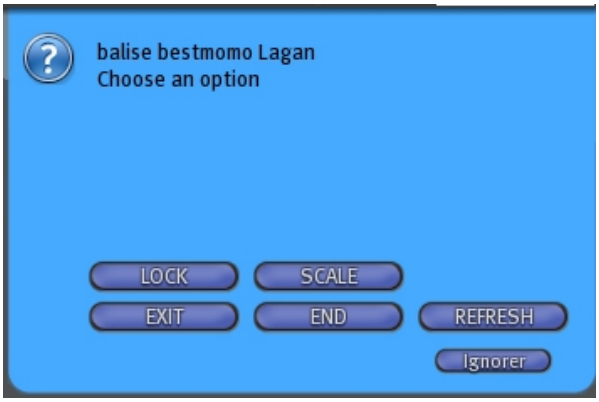
Choisissez l'option « REZ 10m », vous obtenez les éléments de réglage du premier segment de votre trajectoire :



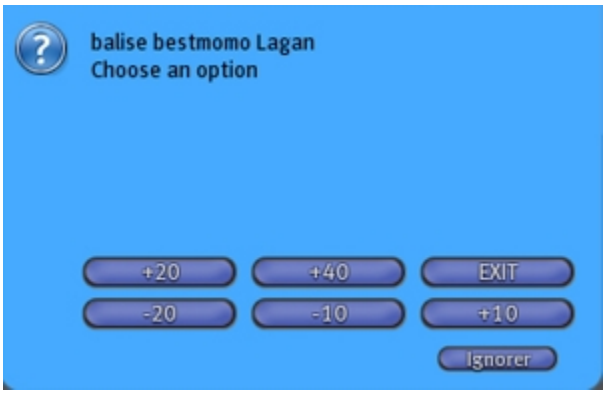
Modifiez la position et la rotation des deux balises pour constater les modifications engendrées :



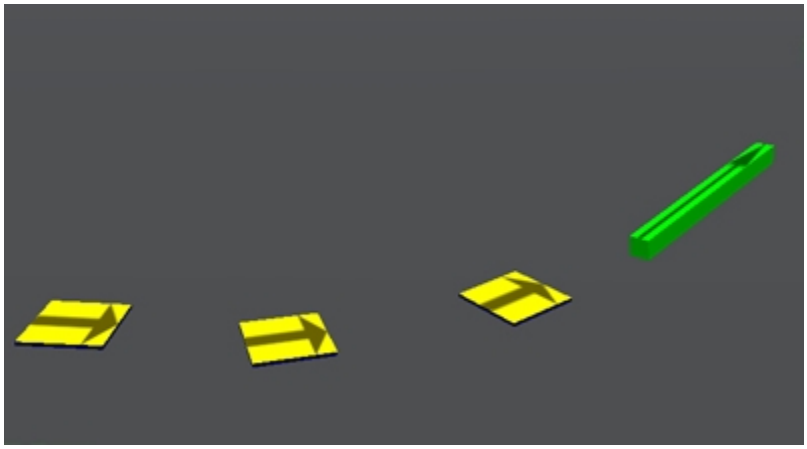
Les bornes intermédiaires s'adaptent à ces modifications et matérialisent la trajectoire obtenue. Cliquez sur l'une des balises, vous obtenez ce menu :



Choisissez l'option « SCALE », vous obtenez ce deuxième menu :

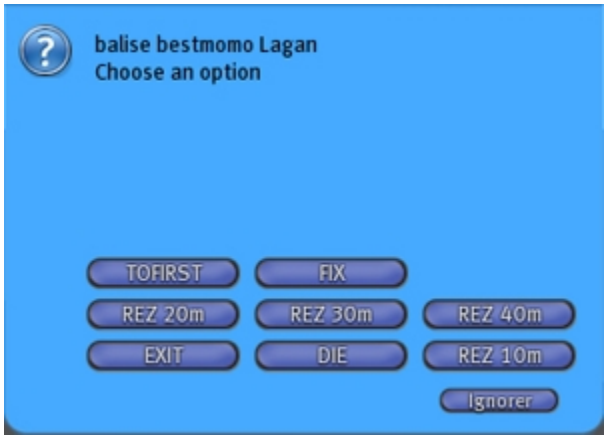


Modifiez la longueur de la balise avec ces boutons pour constater les effets obtenus :

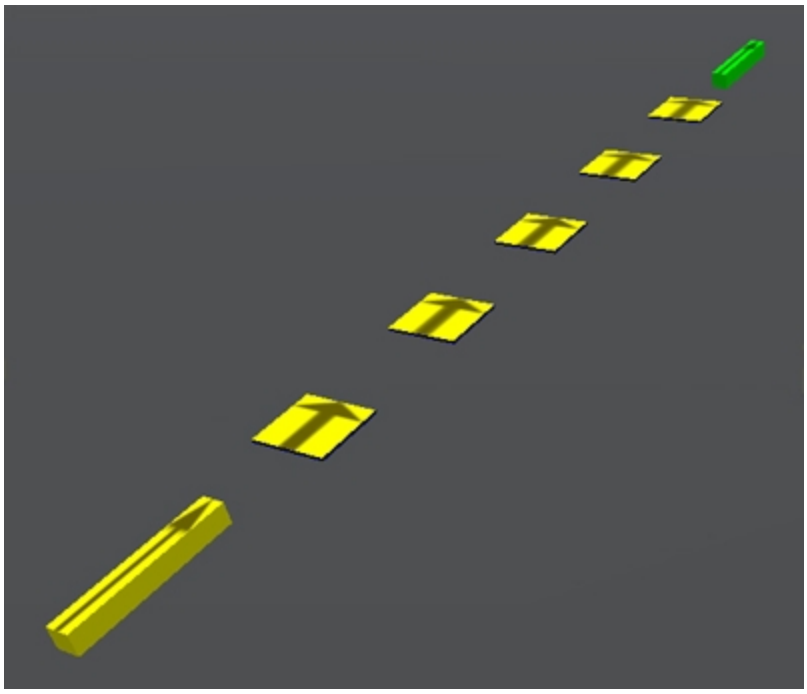


Vous pouvez ainsi ajuster la courbure de la trajectoire. Cliquez sur la deuxième balise et choisissez l'option « END » :

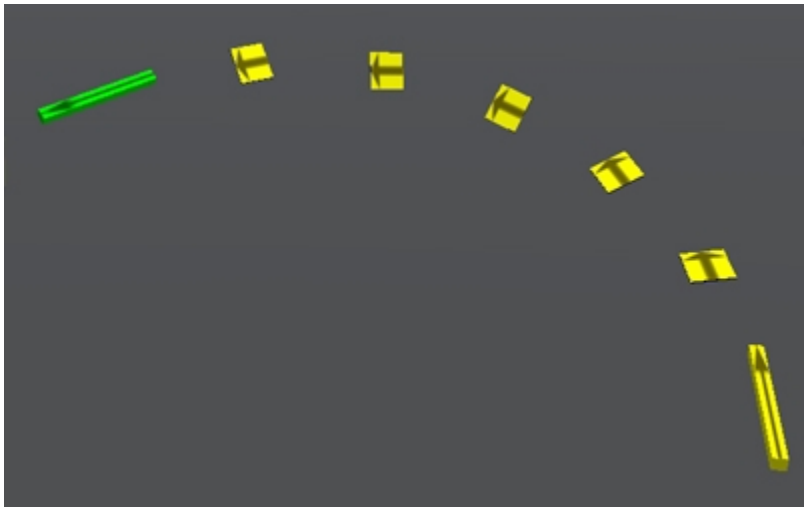
Les bornes intermédiaires disparaissent et les balises changent de couleur pour indiquer leur modification fonctionnelle. Cliquez sur la balise bleue :



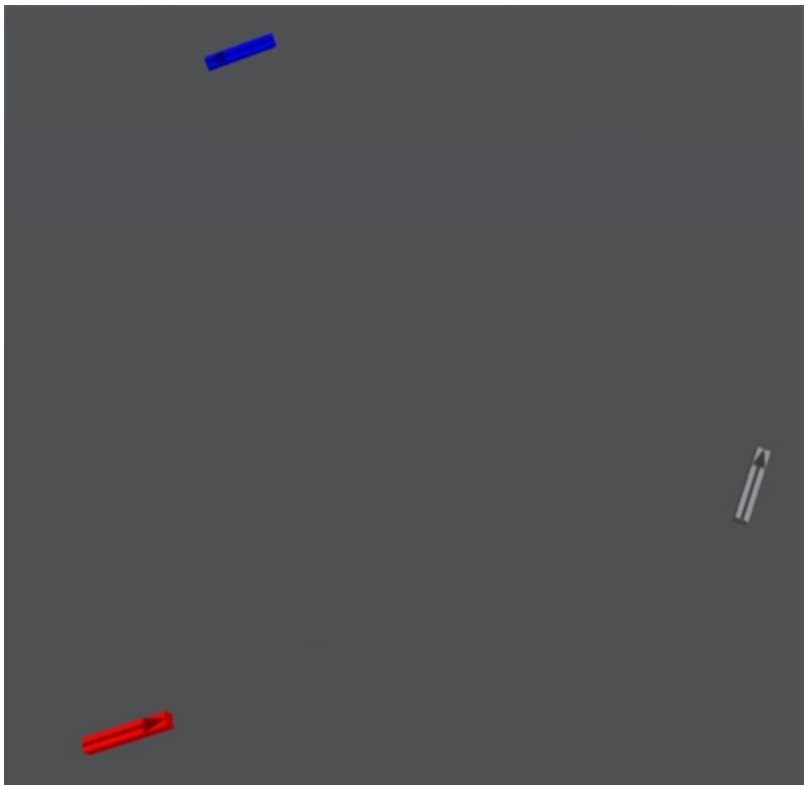
Créez un nouveau segment en choisissant l'option « REZ 10m » :



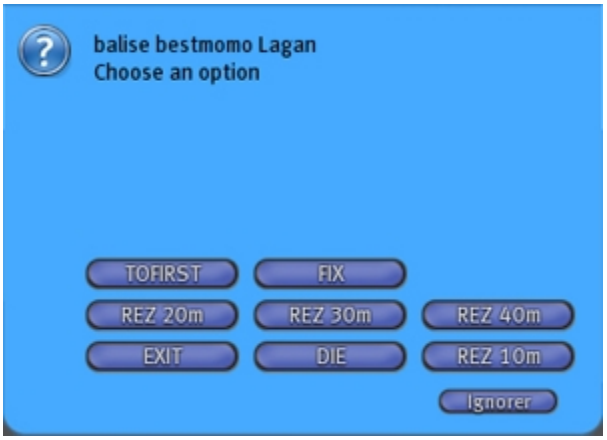
Vous obtenez pratiquement la même chose qu'au départ mais avec une balise initiale jaune. Cette couleur de balise indique qu'on ne peut changer que sa longueur. En effet elle sert de référence au segment précédent pour la position et la rotation ! Incurvez votre segment pour obtenir quelque chose qui ressemble à ça :



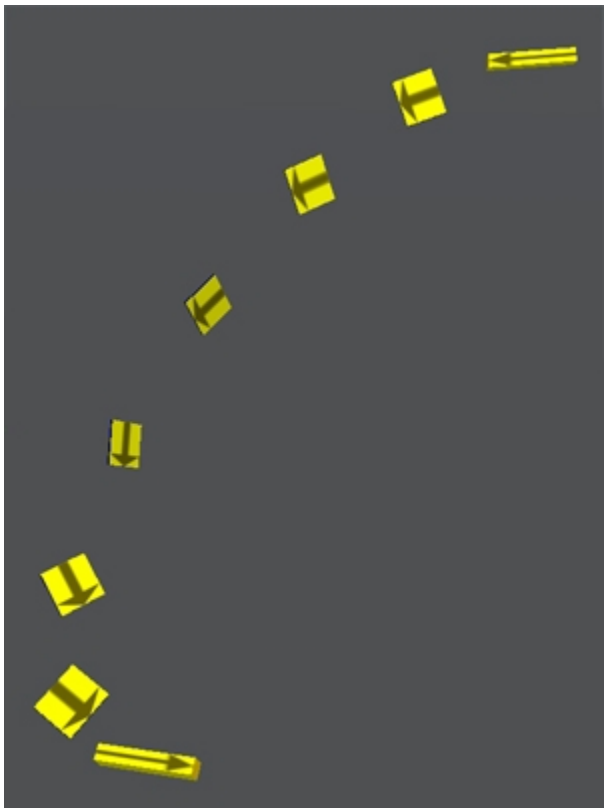
Cliquez sur la balise verte et choisissez l'option « END ». Vous avez maintenant 3 balises :



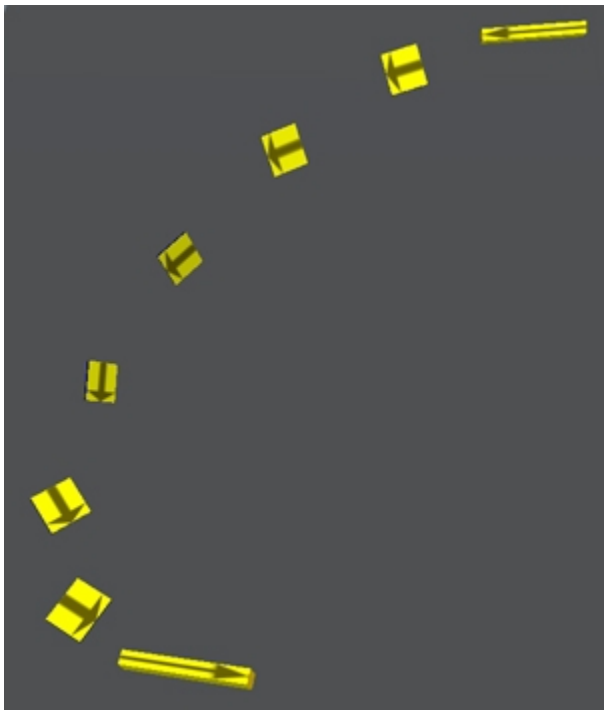
La deuxième balise est devenue grise, ce qui signifie qu'on ne peut plus la modifier. Cliquez sur la balise bleue :



Sélectionnez l'option « TOFIRST » :



Des bornes se créent entre la dernière balise et la première. Ajustez les longueurs si nécessaire pour adoucir la courbure :

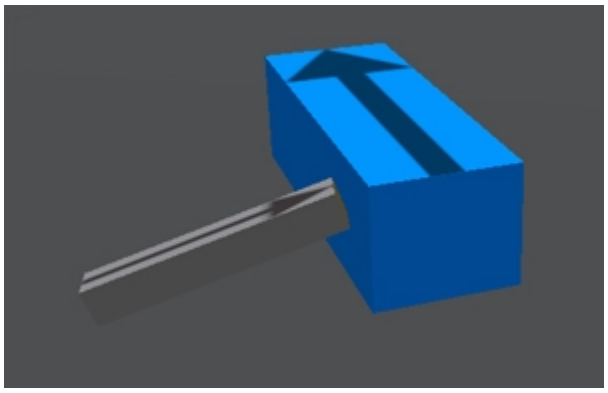


Cliquez alors sur la dernière balise (la première du dernier segment) et choisissez l'option « END » :

Toutes les balises deviennent grises. La trajectoire est terminée. Cliquez sur n'importe quelle balise et choisissez l'option « FIX ». Les balises disparaissent. Ne vous inquiétez pas elle sont toujours là mais invisibles ! Pour la suite de cette prise en main sélectionnez les et faites les réapparaître en mettant leur transparence à 0.

Détection de la trajectoire

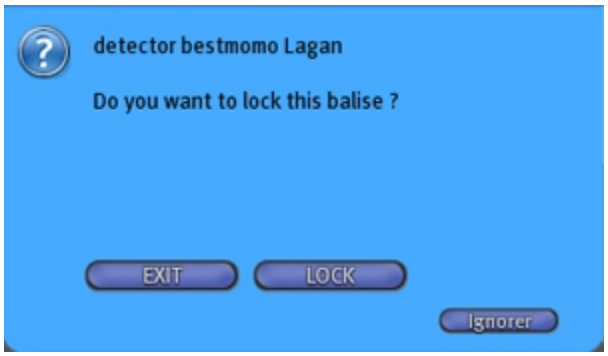
Dans cette deuxième phase nous allons utiliser l'objet « detector » qui ressemble à une grosse balise. Posez-le près d'une de vos trois balises :



Si votre detector est à moins de 5 mètres de la balise il la détecte et vous donne des informations dans le chat :

[8:33] detector: Detector version V 1.0, memory free : 34226 bytes
[8:33] detector: Scanning for balises. Range = 5 meters.
[8:33] detector: I've found a balise type I, I check if it is a closed path
[8:33] detector: The path is closed. I get now path informations, please wait.
[8:33] detector: Position balise 1 : <3.318230, 237.432100, 981.301000>
[8:33] detector: Position balise 2 : <2.765048, 227.407300, 981.837100>
[8:33] detector: Position balise 3 : <13.026790, 231.988100, 984.478800>
[8:33] detector: Length 1 : 11.065110
[8:33] detector: Length 2 : 15.491770
[8:33] detector: Length 3 : 15.412530
[8:33] detector: Total length : 41.969400
[8:33] detector: Path informations are now saved.

On se rend compte qu'il a bien détecté les 3 balises et enregistré leurs coordonnées. Il a aussi calculé les longueurs des segments. Vous obtenez un menu qui vous propose de verrouiller le detector sur la balise :

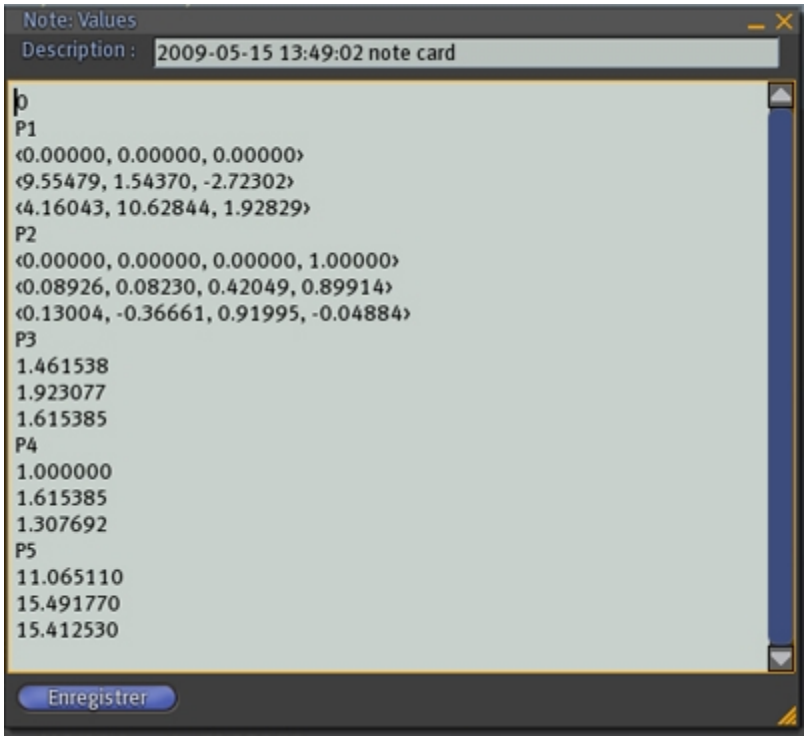


Choisissez l'option « LOCK ». Le detector s'aligne sur la balise et donne de nouvelles informations dans le chat :

[8:37] :::::::::::::: begin notecard ::::::::::::::::::::
0
P1
<0.00000, 0.00000, 0.00000>
<9.55479, 1.54370, -2.72302>
<4.16043, 10.62844, 1.92829>
P2
<0.00000, 0.00000, 0.00000, 1.00000>
<0.08926, 0.08230, 0.42049, 0.89914>
<0.13004, -0.36661, 0.91995, -0.04884>

P3
1.461538
1.923077
1.615385
P4
1.000000
1.615385
1.307692
P5
11.065110
15.491770
15.412530
[8:37] ::::::::::::::: end notecard ::::::::::::::::::::
[8:37] detector: Copy all the text above in a notecard called "Values" and save it
[8:37] detector: Work done... I can die...

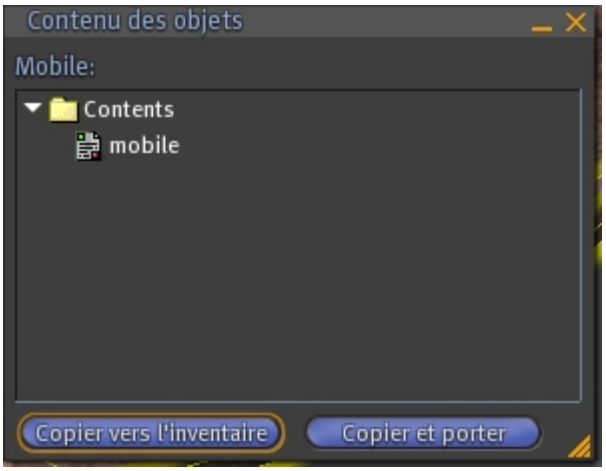
Au bout de quelques secondes il disparaît. Créez alors une notecard que vous nommez « Values » avec les valeurs données :



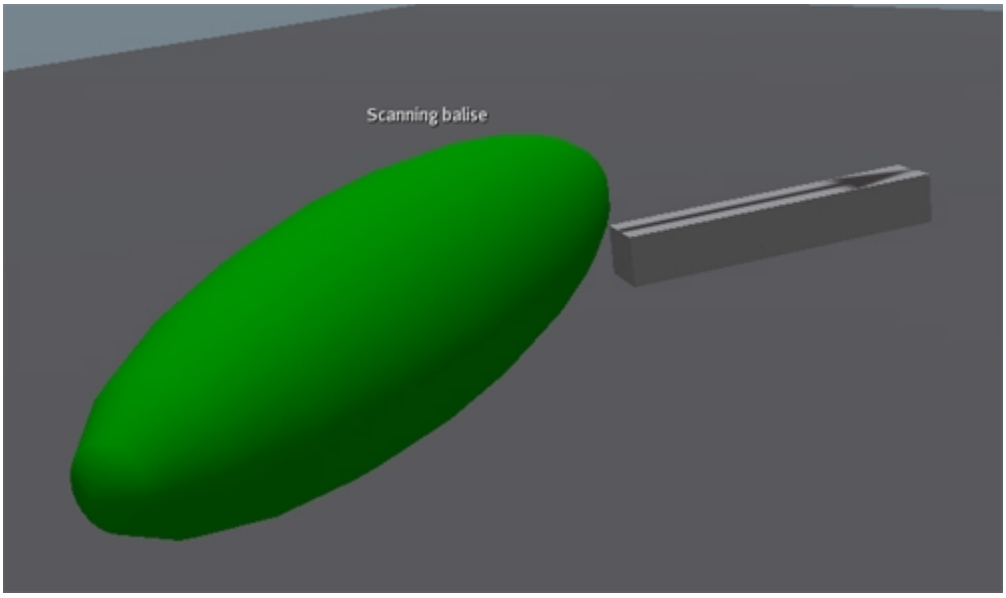
Ces valeurs correspondent aux positions, rotations, longueurs des balises et à la longueur des segments de trajectoire.

Paramétrage du mobile

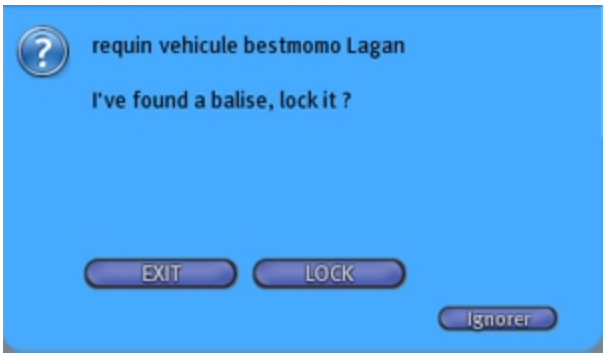
Créez ou utilisez un mobile de votre choix. Le seul critère tient dans le nombre de prims utilisés qui ne doit pas dépasser 32, y compris les passagers éventuels. Le sens de déplacement se situe sur l'axe X local du mobile. Glissez le script « mobile » dans l'inventaire :



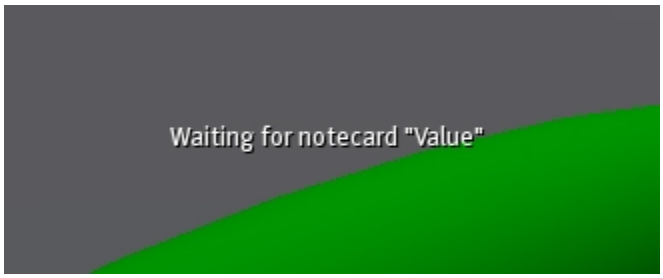
Le mobile schématisé par l'ovoïde vert se place alors en état de recherche de balise (5 mètres), ce qui est indiqué dans le texte flottant :



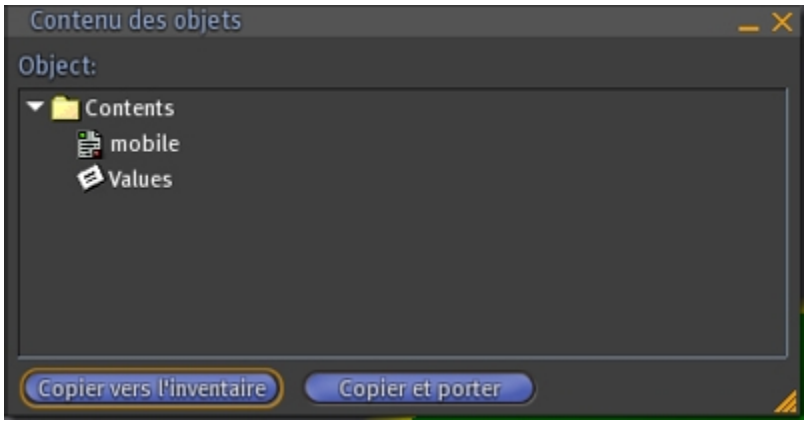
Dès qu'une balise est trouvée, ce menu apparaît :



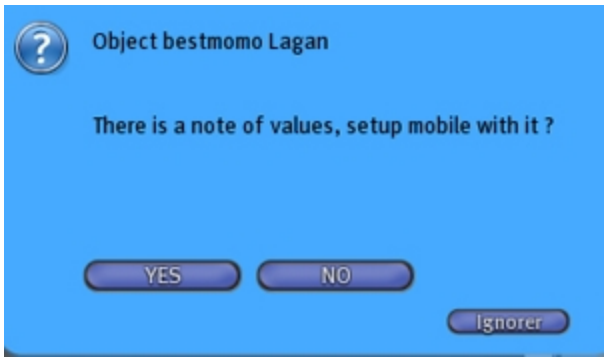
Choisissez « LOCK » pour verrouiller le mobile sur la balise. Le texte flottant se transforme :



Glissez dans l'inventaire du mobile la notecard « Values » que vous avez créée précédemment :



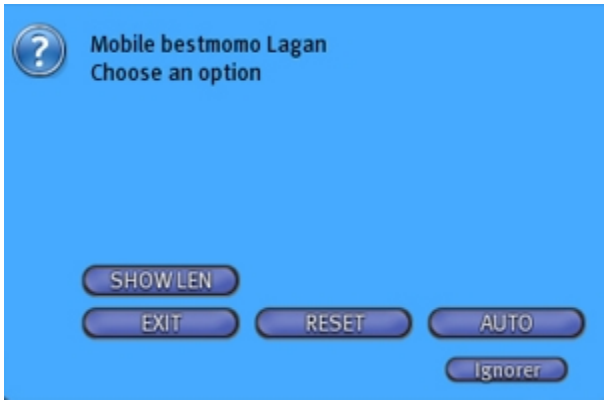
Vous obtenez ce menu :



Choisissez « YES », des infos sont transmises dans le Chat :

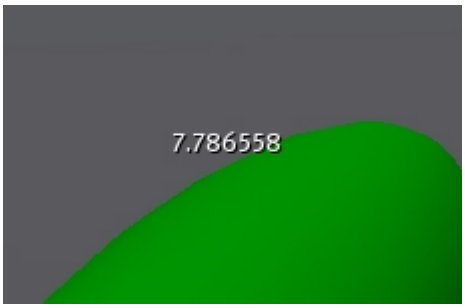
```
[9:43]  Mobile: Reading Notecard...
[9:43]  Mobile: 0
[9:43]  Mobile: P1
[9:43]  Mobile: <0.00000, 0.00000, 0.00000>
[9:43]  Mobile: <9.55479, 1.54370, -2.72302>
[9:43]  Mobile: <4.16043, 10.62844, 1.92829>
[9:43]  Mobile: P2
[9:43]  Mobile: <0.00000, 0.00000, 0.00000, 1.00000>
[9:43]  Mobile: <0.08926, 0.08230, 0.42049, 0.89914>
[9:43]  Mobile: <0.13004, -0.36661, 0.91995, -0.04884>
[9:43]  Mobile: P3
[9:43]  Mobile: 1.461538
[9:43]  Mobile: 1.923077
[9:43]  Mobile: 1.615385
[9:43]  Mobile: P4
[9:43]  Mobile: 1.000000
[9:43]  Mobile: 1.615385
[9:43]  Mobile: 1.307692
[9:43]  Mobile: P5
[9:43]  Mobile: 11.065110
[9:43]  Mobile: 15.491770
[9:43]  Mobile: 15.412530
[9:43]  Mobile:
[9:43]  Mobile: Path informations are now saved.
[9:43]  Mobile: Mobile ready !
[9:43]  Mobile: Memory free : 21318 bytes
```

La notecard a été lue avec succès et le mobile est prêt ! Cliquez sur le mobile, vous obtenez le menu suivant :



Choisissez « SHOWLEN ». Cette option permet d'afficher la position du mobile sur la trajectoire au niveau du texte flottant. L'affichage se fera dès la mise en marche du mobile. Pour faire avancer le mobile portez le HUD :

Commencez par cliquer sur le bouton « S » pour synchroniser le mobile avec le HUD. Ensuite cliquez sur le bouton triangulaire supérieur qui permet de faire avancer le mobile en marche avant. Le mobile démarre et affiche dans son texte flottant sa position sur la trajectoire en mètres depuis l'origine :

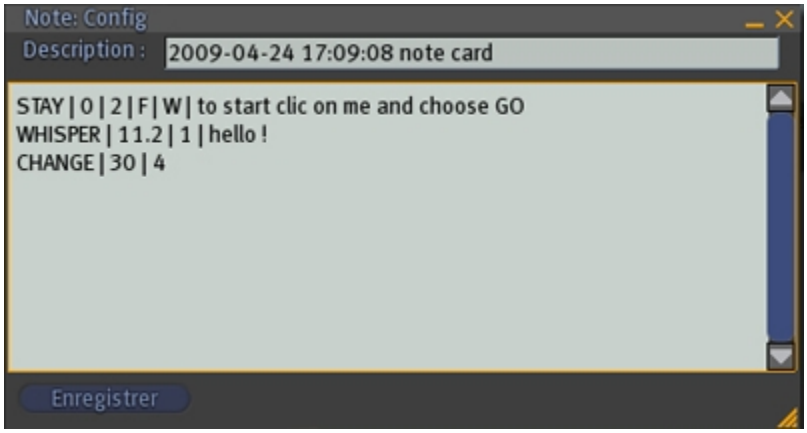


Le bouton « STOP » sert à arrêter le mobile. Le bouton triangulaire inférieur pour la marche arrière. Le changement de sens ne peut s'effectuer qu'après un arrêt. Le changement de vitesse peut s'effectuer en marche.

Ajoutez maintenant les scripts « sequencer » et « syntaxor » dans l'inventaire du mobile :

Créez une notecard que vous nommez « Config » avec les éléments suivants :

```
STAY | 0 | 2 | F | W | to start clic on me and choose GO
WHISPER | 11.2 | 1 | hello !
CHANGE | 30 | 4
```



Si votre trajectoire a une longueur inférieure à 30 mètres diminuez en conséquence le premier paramètre de la commande CHANGE. Glissez la notecard dans l'inventaire du mobile :

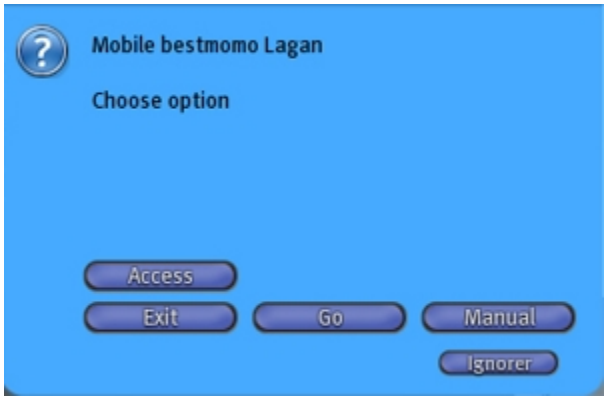
Vous obtenez des informations au niveau du Chat :

```
[10:16]  Mobile: Reading Notecard...
[10:16]  Mobile: STAY | 0 | 2 | F | W | to start clic on me and choose GO
[10:16]  Mobile: WHISPER | 11.2 | 1 | hello !
[10:16]  Mobile: CHANGE | 30 | 4
[10:16]  Mobile: Config Notecard read. Waiting for mobile message AUTO...
```

Le script "sequencer" a bien lu la notecard qui a été validée par le script "syntaxor". Cliquez alors sur le mobile et passez en mode automatique en cliquant sur le bouton « AUTO ». Vous obtenez de nouvelles informations au niveau du Chat :

```
[10:20]  Mobile: Final check..
[10:20]  Mobile: Initializing position...
[10:20]  Mobile: Final check ok !
[10:20]  Mobile: Sequencer in action
[10:21]  Mobile chuchote : to start clic on me and choose GO
```

Un dernier contrôle de validité a été effectué par le script "syntaxor". Ensuite le mobile est positionné en fonction de sa position actuelle. Il finit par s'arrêter au point 0 de la trajectoire (commande STAY). Maintenant le mode automatique est opérationnel. Le mobile obéit au script « sequencer » qui lui-même est paramétré par la notecard « Config ». Le HUD est devenu inactif. Si vous cliquez sur le mobile lorsqu'il est à l'arrêt vous obtenez le menu suivant :



Vous pouvez revenir en mode manuel en choisissant l'option « Manual », gérer les droits d'accès au menu en choisissant « Access », ou démarrer le mobile en choisissant « Go ».

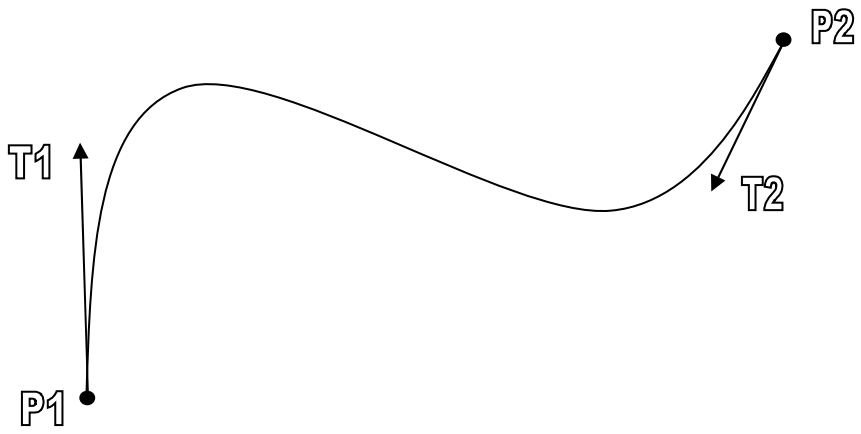
Nous sommes arrivés à la fin de cette prise en main. La suite de ce manuel va vous donner des informations plus précises sur les fonctionnalités que nous venons de survoler.

La Trajectoire

Le système permet de définir une trajectoire quelconque de la ligne droite aux courbes évoluées grâce à une approche mathématique performante.

La spline cubique d'Hermite

D'un point de vue purement mathématique une spline est une fonction définie par morceaux par des polynômes. La spline cubique d'Hermite (en mémoire au célèbre mathématicien Français Charles Hermite) est une spline de degré 3. Visuellement il s'agit d'une courbe contrôlée par deux points et deux tangentes en ces points. Les points intermédiaires sont interpolés entre les deux points de contrôle :



La courbe est totalement contrôlée à l'aide du positionnement des points de contrôle (P1 et P2) et de la direction et la valeur des vecteurs tangents (T1 et T2). Les habitués des applications DAO doivent reconnaître les fonctionnalités des courbes de Bezier.

Vocabulaire

Trajectoire (Path)

Parcours total qui peut être ouvert ou fermé.

Trajectoire fixée

Une trajectoire est fixée lorsque toutes les balises qui la composent sont devenues passives (plus de script). c'est l'état définitif de la trajectoire sans possibilité de retour en arrière. Chaque balise contient dans sa description les clés des balises encadrantes et les valeurs des tangentes des segments encadrants.

Segment

Portion de trajectoire située entre deux balises.

Balise

Prim passive jalonnant la trajectoire délimitant les segments. Les balises mémorisent les clés des balises encadrantes ainsi que la valeur des tangentes des segments adjacents.

Borne

Primitive généré par les balises pour matérialiser un segment et permettre son réglage.

Mobile

Objet "physique" ayant la capacité mémoriser les informations d'une trajectoire, à la parcourir, et à obéir à des commandes simples.

Detector

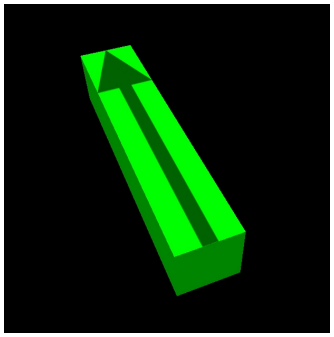
Prim active capable de lire les informations des balises et de les lister dans le Chat.

Code de couleur des balises

Les balises se différencient par leur fonction. Un code de couleurs permet à tout moment de connaître les possibilités de traitement d'une balise.

Balise verte

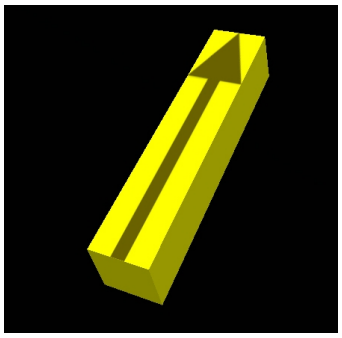
Une balise verte peut subir une translation, une rotation ou un redimensionnement de sa valeur x :



Translation
Rotation
Scale X

Balise jaune

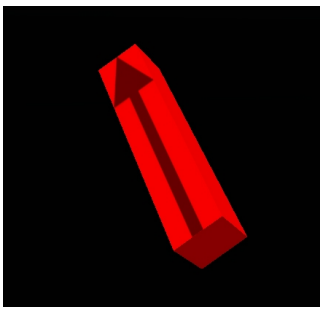
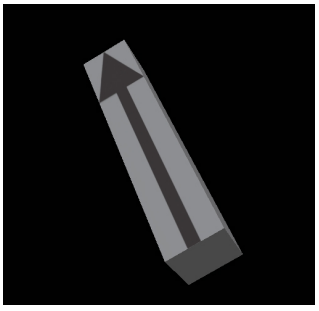
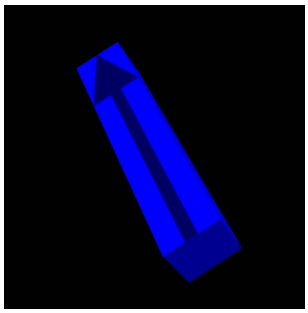
Une balise jaune peut subir seulement un redimensionnement de sa valeur x :



Scale X

Balise rouge, grise ou bleue

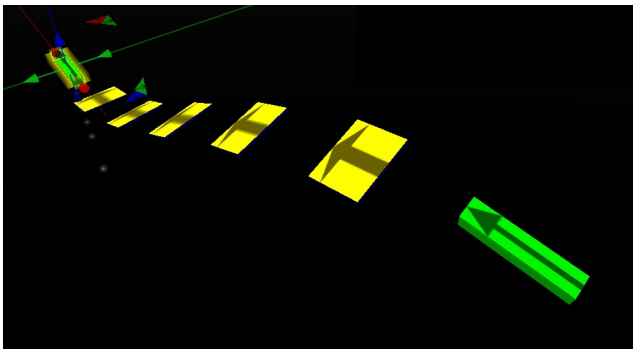
Les autres balises ne peuvent pas être modifiées :

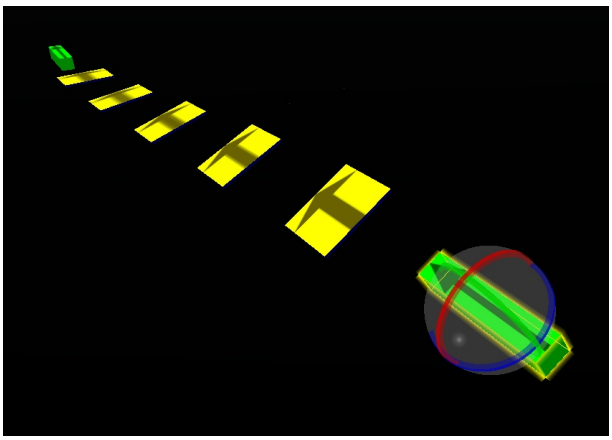




Aucune modification

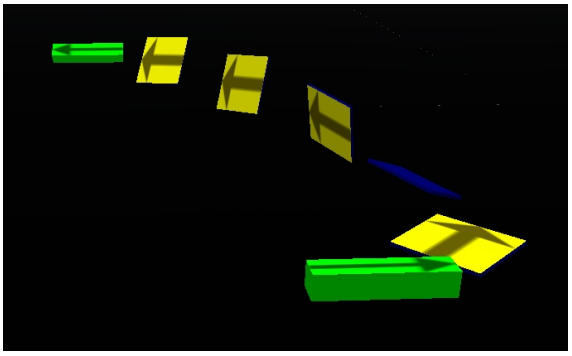
Principe de réglage d'un segment

Un segment est l'élément de trajectoire situé entre deux balises, ce sont donc ces balises qui déterminent la longueur et la forme du segment. Il est possible d'effectuer des translations et rotations de ces balises, ainsi que des modification de leur longueur X. Des bornes sont générées pendant le réglage par l'une ou l'autre balise pour matérialiser visuellement le segment. La valeur du vecteur tangent est matérialisée par la longueur x de chaque balise. Voici deux visuels d'un segment en cours de réglage :





Il y a une certaine liberté de positionnement des balises mais certaines situations extrêmes peuvent générer des aberrations dans la rotation des bornes comme dans l'exemple suivant :



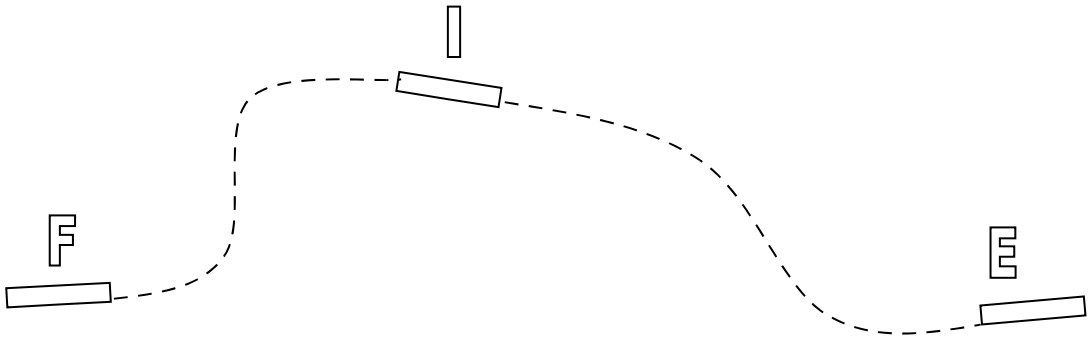
Dans ce cas l'angle est trop resserré et provoque une anomalie dans le calcul de la rotation de la deuxième borne. La solution dans ce cas consiste soit à allonger la tangente en jouant sur la valeur x de la dimension de la première balise de réglage ou de modifier sa rotation. Il convient donc de respecter ces limitations en surveillant le comportement des bornes.

Types de trajectoire et repérage

Une trajectoire est orientée et possède une origine et une extrémité. On se repère sur une trajectoire avec la distance par rapport à l'origine. Par exemple on peut avoir un point situé à 82,24 mètres de l'origine. Cette notion est importante pour la gestion du comportement du mobile sur la trajectoire. Bien qu'étant continue la trajectoire est divisée en petit segments lorsqu'elle est traitée par le mobile. La granularité a été fixée à environ 40 centimètres après un certain nombre d'essais. C'est la valeur qui est apparue la plus pertinente pour concilier la précision de déplacement du mobile et la charge du serveur.

Trajectoire ouverte

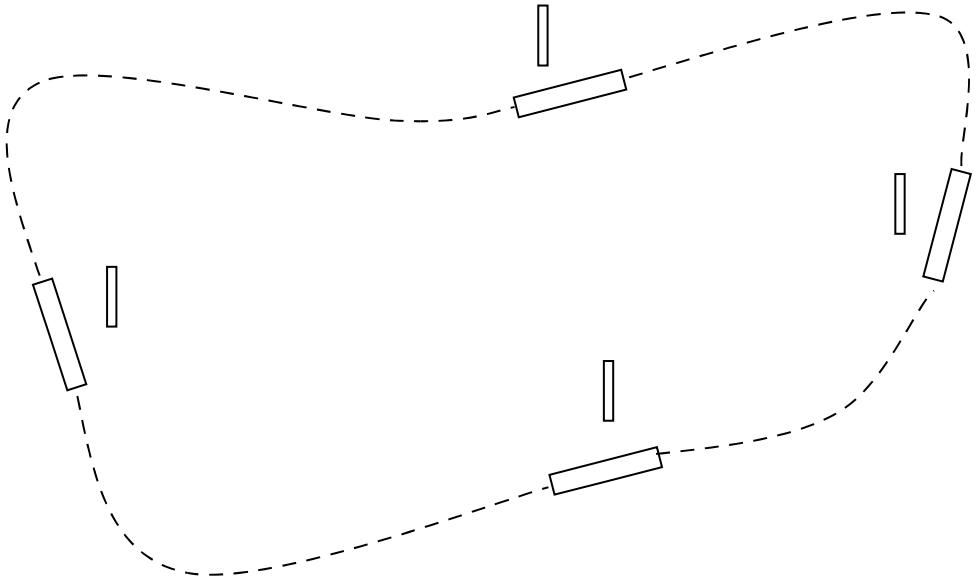
Ce type de trajectoire comporte une balise de départ (type F) et une balise finale (type E). Elle peut comporter des balises intermédiaires (type I). Sur ce type de trajectoire le mobile ne peut faire que des allers-retours.



Dans ce type de trajectoire le repérage d'un point est facile parce que origine et extrémité sont bien distinctes.

Trajectoire fermée

Ce type de trajectoire ne comporte que des balises intermédiaires (type I), avec un minimum de 2. Le mobile peut parcourir ce type de trajectoire, dans un sens ou dans l'autre, de façon continue puisqu'il y a un bouclage.

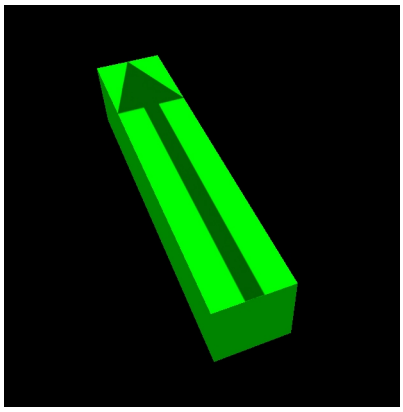


Dans ce type de trajectoire le repérage d'un point est plus délicat parce que origine et extrémité sont confondues. La détermination de l'origine des distances se fait au niveau de la balise qui sert d'initialisation. Ce peut être n'importe quelle balise de la trajectoire.

Mise en place de la trajectoire

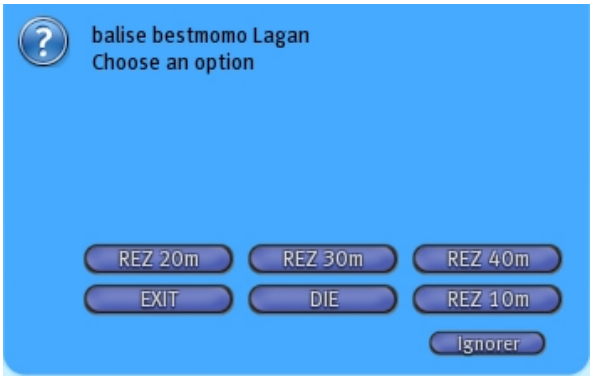
La mise en place de la trajectoire répond à une procédure simple mais précise. Il convient de respecter les différentes étapes.

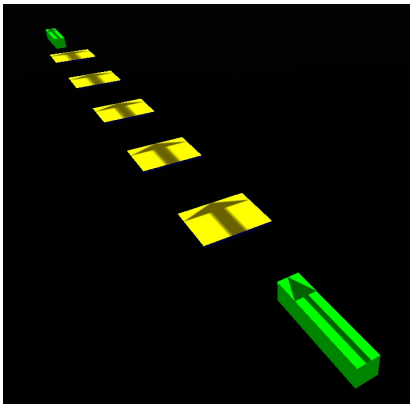
Première balise type F (First)



La première balise posée est la balise de type F (First). C'est une balise un peu particulière parce qu'elle initie la série de balises qui vont constituer la trajectoire. Elle est de couleur verte parce qu'il est possible de modifier sa position, sa rotation et la valeur x de sa dimension (voir code des couleurs évoqué ci dessus). La balise n'étant pas modifiable le changement de la longueur X se fait toujours par menu.

Un clic sur la balise F fait apparaître ce menu :

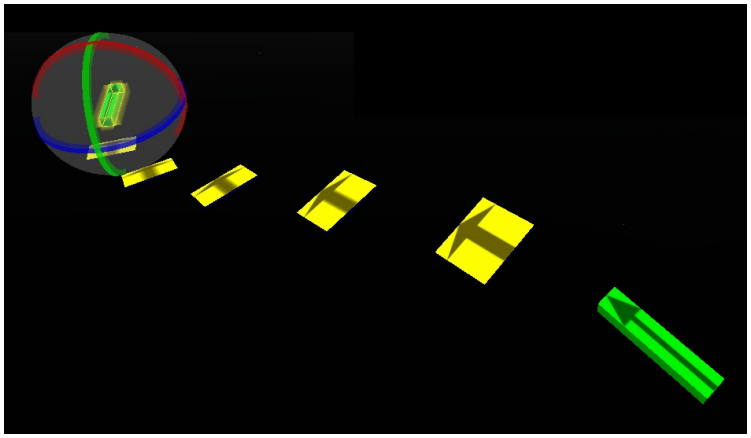




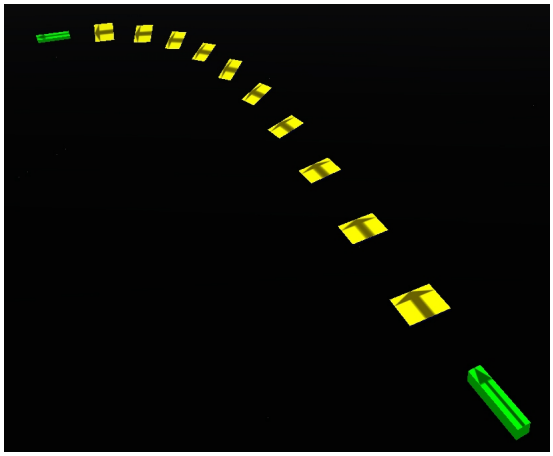
Premier segment

Le premier segment est généré par la première balise (F) en cliquant sur l'un des boutons de Rez (10, 20 ou 30 mètres). Les bornes sont générées et positionnées de façon uniforme sur toute la distance choisie. La deuxième balise est ensuite crée et positionnée à l'extrémité du segment.

En cas d'échec au cours de cette phase, que ce soit au niveau des bornes ou de la deuxième balise, il est possible de supprimer l'ensemble de ce qui a été généré en cliquant sur le bouton "RESET" du menu de la balise F. Il suffit alors de répéter l'opération de génération.

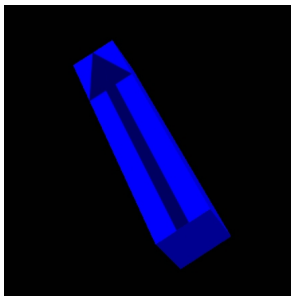


On procède ensuite au réglage du segment en jouant sur la position, la rotation et la dimension x des balises encadrantes.



Si le segment est redimentionné par déplacement des balises il est possible de générer un nouveau jeu de bornes en cliquant sur la deuxième balise et en choisissant l'option "REFRESH" dans le menu. Les bornes présentes sont alors supprimées et un nouveau jeu de bornes est créé et positionné.

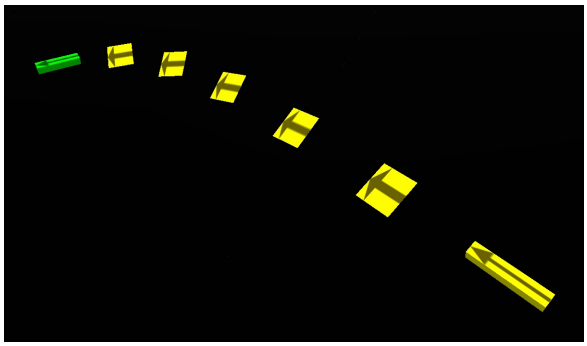
La limite de distance entre deux balises encadrant un segment est de 100 mètres (cette limite est due au fait que la communication entre balises s'effectue au moyen de la fonction **llShout**). Il n'est toutefois pas conseillé d'atteindre cette limite pour des considérations pratiques de réglage.



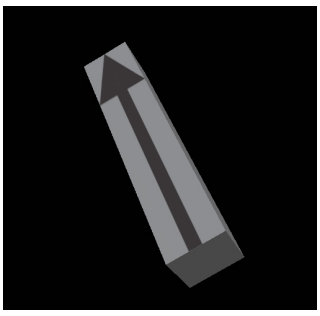
Lorsque le réglage est terminé il faut cliquer sur la deuxième balise et choisir l'option "END" dans le menu affiché. Les bornes sont alors supprimées. La première balise redevient rouge et la seconde bleue.

Segments intermédiaires

La balise terminale d'un segment est la balise initiale du segment suivant. La balise terminale d'un segment est de couleur bleue. Pour générer le nouveau segment il faut cliquer sur la balise bleue et choisir un des boutons de rez. Les bornes et la balise suivante sont générées et positionnée de la même manière que pour le premier segment vu précédemment.



La seule différence est que la balise initiale du segment passe du bleu au jaune. Ce qui signifie qu'elle ne peut pas subir de translation ou de rotation mais seulement un changement de sa dimension x. En effet cette balise étant un point de référence du segment précédent il ne faut plus changer les paramètres de position et de rotation. Par contre la deuxième balise est totalement libre (verte).



Lorsque le réglage est terminé il faut cliquer sur la deuxième balise et choisir l'option "END" dans le menu affiché. Les bornes sont alors supprimées. La première balise devient grise et la seconde bleue.

Une balise située entre deux segments est une balise de type I (Intermediate). Elle ne doit plus subir de modification.

Verrouillage d'une balise

Il est possible au lieu de régler directement la position et la rotation d'une balise libre (verte) de verrouiller celle-ci sur un objet (en principe le mobile qui devra parcourir la trajectoire) en utilisant la commande "LOCK" dans le menu. La procédure est simple : une fois que cette commande a été activée la balise scanne tout objet à moins de 2 mètres et propose de se verrouiller sur le premier détecté. Ensuite toute translation ou rotation au niveau de l'objet est transmise à la balise. Cette possibilité a été créée en pensant à la présence d'une route ou de rails qui doivent être positionnés précisément par rapport au mobile.

Fin et fixation d'une trajectoire

Lorsqu'une trajectoire comporte tous les segments nécessaires il faut la fixer (voir définition au point). Les balises deviennent alors passives et invisibles.

Fin d'une trajectoire ouverte

Une trajectoire ouverte commence par une balise de type F (rouge), peut posséder des balises I (grises), et se termine par une balise O (bleue). La fixation se fait en choisissant l'option "FIX" dans le menu de l'une des balises de la trajectoire. La dernière balise devient alors de type E (End) et marque la limite de la trajectoire.

Fin d'une trajectoire fermée

Une trajectoire fermée doit être bouclée. Il faut donc rattacher la balise terminale du dernier segment à la balise F. Cela se fait simplement en cliquant sur la balise terminale et en choisissant l'option "TO FIRST" dans le menu. Des bornes sont alors générées entre cette balise et la balise d'origine F. Ces deux balises deviennent jaunes parce que vous ne pouvez régler que les tangentes longueur X). Lorsque le réglage est terminé utilisez l'option "END" dans le menu de l'une des balises.

Modification d'une trajectoire fixée

Modifier une balise fixée, que ce soit en translation ou rotation, aura évidemment un effet sur la trajectoire obtenue, mais vous n'avez aucun contrôle visuel sur la modification obtenue.

Le mobile

Le mobile est un objet capable de mémoriser une trajectoire et de se déplacer sur celle-ci.

Lecture et mémorisation de la trajectoire par le detector

J'avais initialement prévu la lecture de la trajectoire par le mobile, mais le script est devenu rapidement bien trop volumineux et j'ai dû créer un nouvel objet intermédiaire : le "detector".

Nous avons vu que les balises sont des éléments passifs (elles ne comportent aucun script) pour des raisons de sollicitation minimale du serveur. Le detector doit donc utiliser un sensor pour repérer une balise, il la reconnait grâce à son nom. Ensuite il lit sa description qui comporte les clés des balises encadrantes, ce qui permet de passer de balise en balise pour lire les valeurs et parcourir ainsi l'ensemble de la trajectoire.

Pour chaque balise le detector mémorise la position, la rotation et les valeurs des tangentes pour les deux segments encadrants, c'est-à-dire tous les éléments qui vont lui permettre de calculer les segments de la trajectoire. Une fois cette mémorisation achevée il propose de se verrouiller sur la balise qu'il a détectée. Les coordonnées des balises sont alors égrennées dans le Chat et il est proposé d'enregistrer ces valeurs dans une notecard nommée "Values".

Par la suite les balises ne servent plus et peuvent être supprimées, sauf celle qui sert de référence initiale, sans conséquence sur le fonctionnement sauf si vous désirez apporter quelques modifications mineures et recréer une notecard. Cette notecard est destinée à être lue par le script du mobile. La seule difficulté réside dans le positionnement du mobile lors de cette initialisation. Il faut qu'il se trouve exactement à la position et la rotation de la balise initiale, c'est pour cette raison qu'il faut la conserver précieusement. Le script du mobile va la détecter et proposer de se verrouiller dessus. Vous n'avez plus qu'à glisser la notecard dans l'inventaire du mobile et la trajectoire sera mémorisée par celui-ci.

Il y a une autre raison pour mémoriser la trajectoire dans une notecard : les balises étant passives et uniquement repérées par leur clé il est évident que vous ne pouvez pas reprendre les balises dans votre inventaire pour les reposer ensuite parce que les clés auront changé et vos balises seront devenues inutilisables. Donc si vous créez une route ou autre que vous comptez commercialiser par la suite ou reposer ailleurs mémorisez la trajectoire dans une notecard et conservez la balise initiale ! Les valeurs de positionnement et de rotation sont mémorisées avec des données relatives à l'emplacement du mobile. Lors d'un rechargement des valeurs à partir de la notecard les données absolues sont recréées à partir de la position et

de la rotation actuelles du mobile. Moralité : si vous déplacez votre parcours réinitialisez le mobile !

Déplacement sur la trajectoire

Le script du mobile

Le déplacement du mobile s'effectue par petits "bonds" d'environ 40 centimètres. Un calcul permet de déterminer la position et la rotation pour chaque étape. Le mobile dispose de 5 valeurs de vitesse : 0,5 m/s, 1 m/s, 2 m/s, 4 m/s, 8 m/s. Il peut se déplacer en marche avant ou en marche arrière. Il peut stopper en n'importe quel point de la trajectoire avec une précision dépendant de la granularitée adoptée pour le déplacement qui a été fixée à environ 40 centimètres. Le démarrage et l'arrêt sont progressifs pour conserver un caractère réaliste.

Le script du mobile assure 3 fonctions essentielles :

1. lire une notecard "Values" contenant des éléments de trajectoire,
2. assurer le déplacement sur la trajectoire mémorisée,
3. obéir aux commandes des scripts de gestion du déplacement (HUD, sequencer...).
4. afficher (en activant l'option correspondante dans le menu) la position sur la trajectoire en mètres par rapport à l'origine.

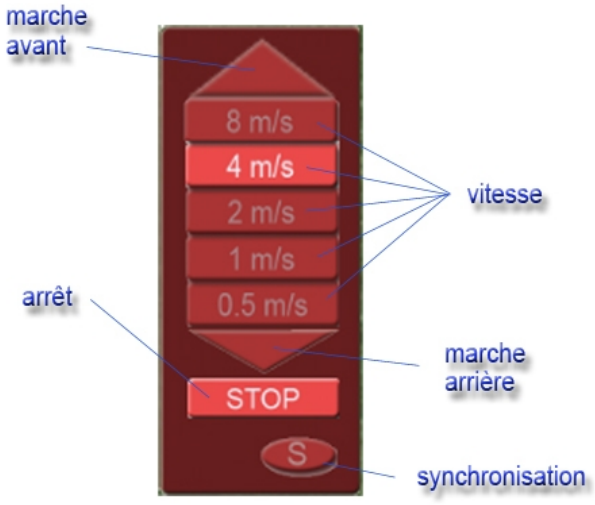
Les commandes reconnues par le mobile sont rudimentaires : avancer, reculer, s'arrêter, changer de vitesse.

Gestion manuelle du déplacement (mode manuel)

Un HUD rassemble toutes les commandes disponibles pour le mobile :

- marche avant,
- marche arrière,
- vitesse : 0,5 m/s, 1 m/s, 2 m/s, 4 m/s, 8 m/s,
- stop,
- synchronisation entre le HUD et le mobile.

L'utilisation est simple : un clic suffit pour activer une option.



Les boutons correspondant aux options actives sont éclairés, les autres éteints. Dans le cas de l'illustration le mobile est à l'arrêt (bouton STOP éclairé) et la vitesse de référence est réglée à 4 m/s.

Gestion automatisée du déplacement (mode auto)

La gestion du mode automatique a impliqué le développement d'un nouveau script, le "sequencer" disposé dans le mobile et chargé d'assurer la commande de ce dernier, en générant les commandes de base du mobile à partir d'un script formé de commandes plus évoluées. Un script complémentaire, le "syntaxor", est chargé de vérifier la validité des commandes utilisées et de certaines fonctions d'initialisation.

Pour passer le mobile en mode automatique il faut avoir déjà ce mobile fonctionnel sur une trajectoire. Ensuite vous glissez les scripts "sequencer" et "syntaxor" dans le mobile accompagnés de la notecard qui définit les commandes. Le script sequencer lit la note et soumet sa syntaxe au script "syntaxor". Si tout se passe correctement un message est là pour vous rassurer, sinon vous recevez des indications sur l'erreur de syntaxe détectée. Ensuite vous cliquez sur le mobile et vous choisissez l'option "Auto" dans le menu qui apparaît. Les deux scripts entrent en communication, si tout se passe bien vous recevez un message de confirmation. Si la communication échoue vous pouvez retrouver le menu du mobile au bout d'un délai d'une minute. Le mobile renseigne le script "sequencer" sur la longueur de la trajectoire et sur sa position actuelle. Le script "sequencer" transmet alors ces informations au script "syntaxor" qui fait une dernière validation des commandes et détermine la commande la plus proche au niveau de la liste et transmet l'index correspondant au "sequencer" pour un cas sans station, sinon il reste en place. Selon l'emplacement du mobile une synchronisation

peut être nécessaire. Vous pouvez ensuite cliquer sur le mobile, vous obtenez maintenant un menu généré par le script "sequencer" aux commandes explicites. Il est toujours possible de revenir en mode "Manuel" en cliquant l'option correspondante dans le menu.

> ☝ Le changement de mode (manuel / auto) ne peut s'effectuer que lorsque le mobile est à l'arrêt.

Mode auto simple

Voici la syntaxe des commandes du sequencer en mode auto simple :

STOP | position | speed | direction (F or B) |time | [W or S] | [text] | [time]
STOPSAY | text | [time] *
STOPWHISPER | text | [time] *
CHANGE | position | speed
SAY | position | speed | text
WHISPER | position | speed | text

La référence au niveau de la trajectoire est la distance à l'origine. Un point particulier est donc repéré par cette distance. Par exemple on peut avoir un point situé à 52,30 mètres de l'origine. Le paramètre "position" correspond à cette valeur. La plupart des commandes comportent ce paramètre pour signaler en quel point particulier de la trajectoire il convient de la prendre en compte.

On a vu que le mobile dispose de plusieurs valeurs de vitesse (0,5, 1, 2, 4, 8 m/s), le paramètre "speed" renseigne cette valeur.

Voyons les différentes commandes disponibles par catégories avec leurs paramètres spécifiques.

Commandes d'arrêt et d'affichage de texte à l'arrêt

Dans le mode auto simple la seule commande d'arrêt est le STOP qui définit une durée d'arrêt au niveau de son quatrième paramètre (time). Le STOP est un arrêt obligatoire à durée programmée.

Le premier paramètre (position) indique l'emplacement de la trajectoire où activer cette commande. Le second paramètre (speed) indique à quelle vitesse le mobile repartira de cet arrêt. Le troisième paramètre (direction) indique la direction du mouvement lorsque le mobile va repartir. Les deux valeurs de ce paramètre sont F (Forward) et B (Backward).

Les paramètres 5 à 7 sont facultatifs. Ils ne servent que si on veut afficher du texte au niveau du Chat. Le cinquième paramètre indique s'il s'agit d'une commande de type "whisper" de

faible portée (W) ou une commande classique de type “say” de portée 20 mètres (S). Le sixième paramètre (text) indique le texte à afficher.

Le septième paramètre (time) ne doit apparaître que si du texte doit être encore affiché pendant l'arrêt. Il convient alors d'utiliser ce paramètre pour indiquer le temps qui doit s'écouler avant l'affichage du texte suivant. Il faut alors utiliser une des commandes STOPSAY ou STOPWHISPER pour définir le second texte à afficher. Si on a besoin d'afficher encore un texte le paramètre (time) des ces commandes est renseigné et une nouvelle commande STOPSAY ou STOPWHISPER est utilisée et ainsi de suite.

Il faut évidemment que la somme des durées entre textes soit inférieure à la durée de l'arrêt défini par la commande STOP.

Commandes intermédiaires

Il peut être nécessaire de changer de vitesse en cours de trajectoire, la commande CHANGE est alors utilisée. Le premier paramètre (position) indique à quel endroit de la trajectoire ce changement doit être réalisé et le second paramètre (speed) renseigne sur la vitesse à appliquer.

Il peut aussi être utile d'afficher un texte dans le Chat en cours de mouvement. Les deux commandes SAY et WHISPER sont alors utilisées. Les deux premiers paramètres sont identiques à ceux de la commande CHANGE, ce qui permet de faire également un changement de vitesse avec ces commandes. Le troisième paramètre (text) indique quel texte doit être affiché.

Exemple de script en mode auto simple

Voici un exemple très simple de script d'un mobile entièrement autonome sur une trajectoire bouclée (à noter qu’un script doit comporter au moins 2 commandes pour être validé par le « syntaxor ») :

```
CHANGE | 10 | 4
STOP | 30 | 2 | F | 5 | W | hello
```

Le mobile tourne indéfiniment avec un changement de vitesse au point 10 m (4 m/s) et un arrêt de 5 secondes au point 30 m avec affichage de faible portée (whisper) du texte “hello”et redémarrage à la vitesse 2 m/s.

Les positions ne sont pas obligées de se suivre selon une progression logique. Le script suivant est accepté par le syntaxor :

```
CHANGE | 10 | 4
CHANGE | 30 | 8
CHANGE | 20 | 1
```

Avec ce script le mobile va à la vitesse de 4 m/s du point 10 m au point 30 m, puis à 8 m/s du point 30 m au point 20 m, il va ensuite à 1 m/s du point 20 m au point 10 mètre. Et le cycle se répète indéfiniment. On se rend compte qu'entre les points 10 m et 20 m il va successivement à 4 m/s et 8 m/s.

Mode auto avec arrêt permanent

Dans ce mode une nouvelle commande apparaît en plus des précédentes qui restent valables, la commande STAY. Cette commande correspond à un arrêt obligatoire et demande une commande externe (bouton « Go » du menu du sequencer) pour que le mobile reparte. Voici la syntaxe de cette commande :

STAY | position | speed | direction (F or B) | [W or S] | [text] | [time]

Comme pour la commande STOP le premier paramètre (position) indique l'emplacement de la trajectoire où activer cette commande. Le second paramètre (speed) indique à quelle vitesse le mobile repartira de cet arrêt. Le troisième paramètre (direction) indique la direction du mouvement lorsque le mobile va repartir. Les deux valeurs de ce paramètre sont F (Forward) et B (Backward).

Les paramètres 4 à 6 sont facultatifs. Ils ne servent que si on veut afficher du texte au niveau du Chat. Le quatrième paramètre indique s'il s'agit d'une commande de type "whisper" de faible portée (W) ou une commande classique de type "say" de portée 20 mètres (S). Le cinquième paramètre (text) indique le texte à afficher.

Le sixième paramètre (time) ne doit apparaître que si du texte doit être encore affiché pendant l'arrêt. Il convient alors d'utiliser ce paramètre pour indiquer le temps qui doit s'écouler avant l'affichage du texte suivant. Il faut alors utiliser une des commandes STOPSAY ou STOPWHISPER pour définir le second texte à afficher. Si on a besoin d'afficher encore un texte le paramètre (time) des ces commandes est renseigné et une nouvelle commande STOPSAY ou STOPWHISPER est utilisée et ainsi de suite.

Il est possible d'utiliser plusieurs fois la commande STAY pour définir plusieurs points d'arrêt.

Exemples de scripts en mode auto avec arrêt permanent

Voici un script de "grand huit" :

```
STAY | 0 | 2 | F | W | to start clic on me and choose GO
WHISPER | 11.2 | 1 | take care
WHISPER | 18.6 | 4 | let's go !
WHISPER | 25.7 | 8 | help !
WHISPER | 30.4 | 4 | still alive...
CHANGE | 40 | 2
```

Une commande STAY est utilisée pour définir un point d'arrêt nécessitant une action manuelle pour repartir. Un texte s'affiche dans le Chat pour renseigner l'utilisateur. Ensuite des changements de vitesses sont effectués en des lieux judicieux du parcours avec un texte pour les utilisateurs.

Voici maintenant un autre script pour une trajectoire ouverte de 31 mètres avec deux arrêts aux extrémités :

STAY | 0 | 4 | F | W | to start clic on me and choose GO
WHISPER | 10 | 4 | look at this nice land
STAY | 31 | 8 | B | W | to start clic on me and choose GO
WHISPER | 20 | 4 | have a nice day !

Avec ce script on définit un arrêt à chacune des extrémités de la trajectoire avec un aller-retour, le retour s'effectuant évidemment en marche arrière.

Voici enfin un script avec beaucoup de texte à délivrer sur une trajectoire bouclée :

STAY | 0 | 2 | F | S | welcome on this line ! | 4
STOPSAY | look at this wonderful land | 4
STOPSAY | to start clic on me and choose GO
CHANGE | 40 | 4
STOP | 80 | 2 | F | 10 | W | look at these trees | 4
STOPWHISPER | look at these cows | 4
STOPWHISPER | look at these houses
CHANGE | 120 | 2

La plupart des erreurs de syntaxe sont signalées par le “syntaxor” au moment de la lecture de la notecard et lors de la synchronisation avec le mobile mais celui-ci ne peut pas tout détecter !

Mode auto avec stations

Dans ce mode apparaît deux nouvelles commandes : STATION et GROUP. Voici la syntaxe de STATION :

STATION | position | speed | direction (F or B) |name | [W or S] | [text] | [time]

La STATION est un arrêt nommé (quatrième paramètre) et sans limite de durée. La STATION n’implique un arrêt du mobile que si cette station est l’objectif de celui-ci, sinon elle est tout simplement ignorée.

☝ Les commandes STAY et STATION ne peuvent pas être utilisées conjointement

Comme pour la commande STOP premier paramètre (position) indique l'emplacement de la trajectoire où activer cette commande. Le second paramètre (speed) indique à quelle vitesse le mobile repartira de cet arrêt. Le troisième paramètre (direction) indique la direction du mouvement lorsque le mobile va repartir. Les deux valeurs de ce paramètre sont F (Forward) et B (Backward).

Les paramètres 5 à 7 sont facultatifs. Ils ne servent que si on veut afficher du texte au niveau du Chat. Le cinquième paramètre indique s'il s'agit d'une commande de type "whisper" de faible portée (W) ou une commande classique de type "say" de portée 20 mètres (S). Le sixième paramètre (text) indique le texte à afficher.

Le septième paramètre (time) ne doit apparaître que si du texte doit être encore affiché pendant l'arrêt. Il convient alors d'utiliser ce paramètre pour indiquer le temps qui doit s'écouler avant l'affichage du texte suivant. Il faut alors utiliser une des commandes STOPSAY ou STOPWHISPER pour définir le second texte à afficher. Si on a besoin d'afficher encore un texte le paramètre (time) des ces commandes est renseigné et une nouvelle commande STOPSAY ou STOPWHISPER est utilisée et ainsi de suite.

La commande STATION implique la présence d'un organe externe de commande avec échange d'informations. L'idée de base est d'obtenir une trajectoire jalonnée de stations avec une borne à chaque station qui permet l'appel du mobile. En retour l'utilisateur peut obtenir un message d'occupation actuelle du mobile en le priant d'attendre ou alors l'arrivée prochaine.

La syntaxe de la commande GROUP est très simple :

GROUP | number

A quoi sert cette commande ? Il peut être utilisé plusieurs trajectoires avec stations sur un même simulateur. Dans ce cas les échanges entre stations et mobile doivent être différenciés. C'est l'objet de cette commande GROUP qui comporte comme seul paramètre un nombre. Evidemment les stations doivent être repérées avec le même numéro de groupe.

Exemples de scripts en mode auto avec STATION

Voici un exemple simple :

```
GROUP | 1
STATION | 0 | 2 | F | alpha | W | welcome on alpha
CHANGE | 40 | 4
STATION | 60 | 4 | F | beta | W | welcome on beta
```

CHANGE | 80 | 2
STATION | 100 | 8 | F | delta | W | welcome on delta

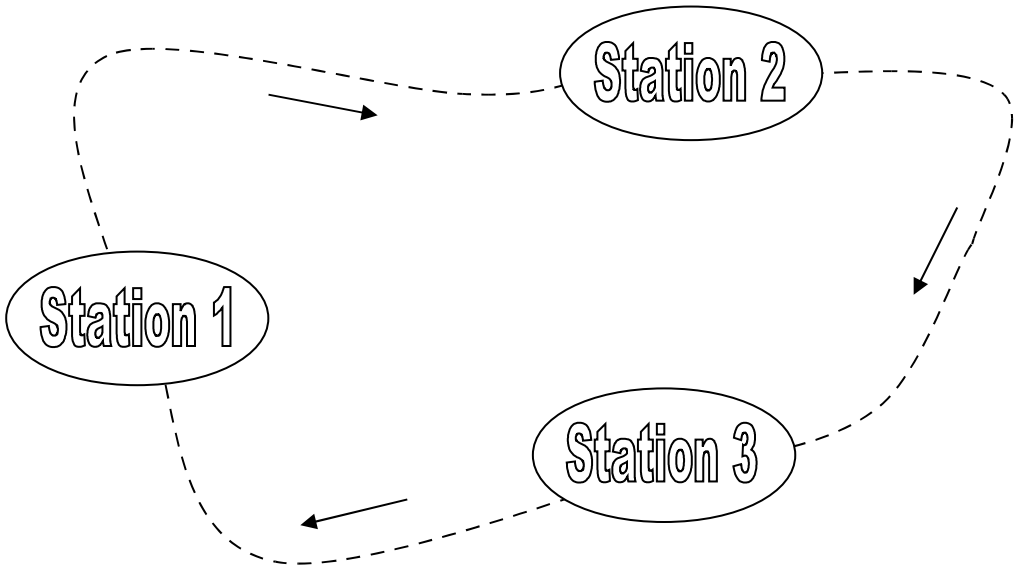
Trois stations, alpha, beta et delta, ont été définies sur une trajectoire bouclée avec des changements de vitesse intermédiaires. A l'initialisation le sequencer laisse le mobile en place, quelle que soit celle-ci. Ensuite il attend une commande du script « station ».

Le script « station » équipe les différentes stations. Son rôle est de servir d'interface entre les utilisateurs et le sequencer. L'utilisateur sollicite la présence du mobile en cliquant sur une borne. Si le mobile est occupé un message d'occupation est renvoyé par le sequencer. Dans le cas contraire une annonce d'arrivée prochaine est donnée. Lorsque le mobile se rend à une station suite à une sollicitation il le fait à sa vitesse maximale en ignorant les arrêts et consignes de vitesses. L'utilisateur occupe le mobile et donne l'ordre de départ en cliquant à nouveau sur la borne. Le mobile se met alors en mouvement en respectant toutes les consignes et en ignorant les stations, sauf évidemment celle de sa destination. Le nom de la station ainsi que son groupe d'appartenance est renseigné dans la description de l'objet qui sert de borne d'appel sous la forme :

Numéro de groupe, nom de station

Il n'est pas possible de combiner des commandes STAY et STATION dans un même script de commandes. En effet cela risquerait de provoquer des confusions parce que la commande STAY est impérative pour le mobile et s'il en rencontrait une en se rendant à une station il serait obligé de s'arrêter.

Une même station peut apparaître deux fois dans un script de sequencer (mais bien entendu avec la même position). Considérez cette trajectoire bouclée :



Le sens normal de circulation est donné par les flèches. Lorsque le mobile est à la station 1 et qu'il veut se rendre à la station 2 il va le faire directement dans le sens normal. Mais s'il veut se rendre à la station 3 il le fera plus rapidement en faisant marche arrière jusqu'à celle-ci. Le même raisonnement est à tenir pour les deux autres stations. Voici la structure de base du script à mettre en œuvre :

GROUP | 1
STATION | 0 | 2 | F | station 1
...
STATION | 20| 2 | F | station 2
...
STATION | 40| 2 | F | station 3
...
STATION | 0 | 2 | B | station 1
...
STATION | 40| 2 | B | station 3
...
STATION | 20| 2 | B | station 2
…

Les stations apparaissent deux fois pour gérer les deux sens de circulation. Le sequencer doit être capable de déterminer le bon morceau de script à exécuter selon le cas. Il est évidemment possible d'imposer un sens de circulation en ne prévoyant que celui-ci dans le script, par exemple pour le sens normal :

GROUP | 1

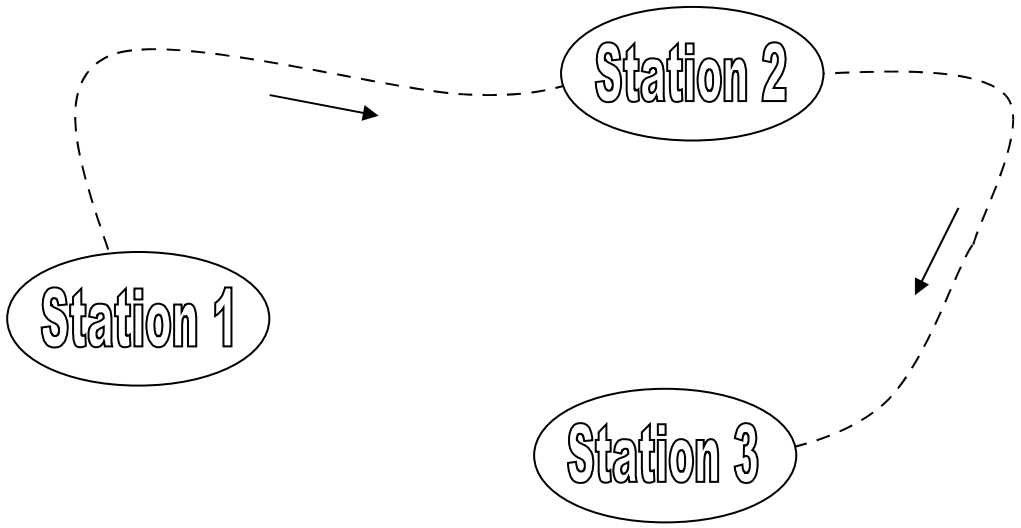
STATION | 0 | 2 | F | station 1
...
STATION | 20| 2 | F | station 2
...
STATION | 40| 2 | F | station 3
...

ou alors pour le sens inverse :

GROUP | 1
STATION | 0 | 2 | B | station 1
...
STATION | 40| 2 | B | station 3
...
STATION | 20| 2 | B | station 2
…

Dans ces deux derniers cas les stations n'apparaissent évidemment qu'une seule fois.

La répétition des stations est par contre indispensable dans le cas d'une trajectoire ouverte :



Dans le cas ci-dessus le script du sequencer doit avoir cette structure :

GROUP | 1
STATION | 0 | 2 | F | station 1
...
STATION | 20| 2 | F | station 2

…
STATION | 40| 2 | B | station 3
…
STATION | 20| 2 | B | station 2
…

Contraintes syntaxiques en mode auto avec STATION

Dans ce mode vous avez un certain nombre de contraintes syntaxiques liée au fonctionnement en navette :

- La première commande (hormis GROUP) doit être une commande STATION,
- Une même STATION ne peut apparaître que 2 fois et avec des sens différents
- Les positions doivent être progressives pour un même sens de circulation,
- Le sens normal doit apparaître en premier,
- Aucun mélange de sens de circulation ne doit apparaître.

Voyons quelques exemples :

```
GROUP | 1
CHANGE | 0 | 4                    Erreur ! La première commande n'est pas STATION !
STATION | 10 | 2 | F | station 1
CHANGE | 20 | 4
STATION | 30| 2 | F | station 2
```

```
GROUP | 1
CHANGE | 0 | 4                    Erreur ! La station "station 1 " apparaît 2 fois avec F !
STATION | 10 | 2 | F | station 1
CHANGE | 20 | 4
STATION | 10| 2 | F | station 1
```

```
GROUP | 1
STATION | 10 | 2 | F | station 1
CHANGE | 30 | 4                    Erreur ! Non progressivité des positions !
STATION | 20| 2 | F | station 2
```

```
GROUP | 1
STATION | 0 | 2 | F | station 1
CHANGE | 10 | 4
STATION | 20| 2 | F | station 2
CHANGE | 30 | 4
STATION | 40| 2 | B | station 3
CHANGE | 30 | 4                    Erreur ! Mélange de F et B !
```

```
STATION | 0 | 2 | F | station 2
```

```
GROUP | 1
STATION | 0 | 2 | B | station 1
CHANGE | 50 | 4
STATION | 20| 2 | F | station 2
```

Erreur ! B apparaît en premier !

```
GROUP | 1
STATION | 50 | 2 | F | station 1
CHANGE | 10 | 4
STATION | 20| 2 | F | station 2
CHANGE | 30 | 4
STATION | 40| 2 | B | station 3
CHANGE | 30 | 4
STATION | 0 | 2 | B | station 2
```

Script correct !