1 Introduction

1.1 Configuration requise

CeeBot requiert un ordinateur moderne et puissant pour fonctionner. La carte graphique 3D a une énorme importance sur les performances. Cela pose parfois des problèmes avec les ordinateurs portables.

- Processeur 300 MHz
- 64 Mb RAM
- Carte graphique 3D avec 16 Mb RAM
- 100 Mb d'espace libre sur le disque dur
- Windows® XP, 2000, ME, 98 ou 95

Les cartes graphiques suivantes sont recommandées :

- nVidia GeForce 2, 3 ou 4
- Matrox G400, G450 ou Parhelia

Les cartes 3dfx Voodoo ne sont pas officiellement supportées, bien que cela puisse fonctionner (sans garantie).

1.2 Installation

- Insérez le CD-ROM CeeBot-A dans votre lecteur

Normalement, l'*autorun* s'active automatiquement après une dizaine de secondes. Cliquez simplement sur « **Installer** ». Si ce n'est pas le cas, procédez comme suit :

- Double-cliquez sur **Poste de travail**.
- Double-cliquez sur **(D:)** ou sur la lettre correspondant au lecteur de CD-ROM.
- Double-cliquez sur **install.exe**.

Si ce n'est pas déjà fait, le programme vous demande d'installer DirectX® 8a.

Si vous avez déjà installé une version précédente de CeeBot, une nouvelle installation dans le même dossier détruira toutes vos sauvegardes.

1.3 Désinstallation

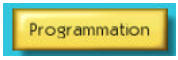
- Double-cliquez sur **Poste de travail**.
- Double-cliquez sur **Panneau de configuration**.
- Double-cliquez sur **Ajout/Suppression de programmes**.
- Double-cliquez sur **CeeBot** dans la liste.

2 Premiers pas

Dans ce chapitre, nous allons effectuer le deuxième exercice, pas à pas. Les explications détaillées viennent dans les chapitres suivants. Il s'agit juste ici d'un premier contact, pour vous familiariser avec CeeBot.

La première question posée par CeeBot est le nom du joueur. Il est important de donner son nom, car chaque utilisateur verra sa progression automatiquement sauvegardée. Donc, tapez simplement votre prénom ou votre pseudo la première fois. Par la suite, il suffira de choisir dans la liste. Lorsque c'est fait, cliquez « **D'accord** ».

Cliquez ensuite sur ce bouton :



CeeBot affiche maintenant la liste des exercices disponibles. La partie gauche correspond aux chapitres, et celle de droite aux exercices du chapitre sélectionné.

Un « vu » indique un chapitre ou un exercice réussi.

Les exercices sont classés par difficulté croissante. Il est donc conseillé de les faire dans l'ordre, bien que ce ne soit pas obligatoire.

Après avoir choisi « **1: Base** » et « **2: Le droit chemin** », cliquez sur le bouton « **Jouer** ».

Le message vert en haut de l'écran vous invite à appuyer sur « **F1** » pour consulter votre ***Sat*Com**, qui vous donnera toutes les instructions sur l'exercice en cours.

Vous pouvez aussi appuyer sur ce bouton, en bas au milieu de l'écran :

La lecture attentive de cet écran vous explique ce que vous devez faire. Lorsque vous avez compris, cliquez sur le bouton en bas à gauche pour sortir.

Si nécessaire, vous pouvez revenir dans cet écran en tout temps, en appuyant simplement sur « **F1** ».

Le robot est actuellement déjà sélectionné, comme le confirme le cadre orange autour de son icône, en haut à gauche. Si ce n'était pas le cas, il suffirait de cliquer dessus avec le bouton de gauche, ou directement sur le robot dans l'écran.

Si ce n'est pas déjà fait, sélectionnez le programme numéro 1 en haut de la liste. Chaque robot peut recevoir 10 programmes, ce qui peut être utile pour faire des essais. Tous ces programmes sont automatiquement sauvegardés lorsque vous quittez l'exercice.

Le programme numéro 4 contient la **solution**. Essayez de ne pas le regarder pour l'instant !

Pour entrer dans l'éditeur de programme, cliquez l'icône **{..}**.

L'écran doit alors montrer un squelette de programme. Le curseur est directement positionné là où vous devez entrer les instructions du programme.

Ne touchez pas au texte `extern void object::Go()`, ni aux accolades.

L'exercice consiste à tracer un trait rouge de 20 mètres, jusqu'à la croix bleue. Pour cela, entrez une première instruction :

```
pendown(Red);
```

Attention au « R » majuscule de « Red ». Tout le reste est en minuscules. N'oubliez pas le point-virgule qui termine l'instruction. Cette instruction abaisse le crayon virtuel du robot, en choisissant la couleur rouge.

Il faut maintenant entrer la deuxième instruction, de préférence à la ligne suivante :

```
move(20);
```

Cette instruction demande au robot d'avancer tout droit de 20 mètres.

Cliquez alors sur le bouton « D'accord ». Si vous avez fait une faute, les caractères erronés sont mis en évidence en bleu, et un message d'erreur apparaît en bas.

Dans l'exemple ci-contre, c'est le point-virgule après la première instruction qui manque.

Lorsque tout est juste, l'éditeur se ferme.

Cliquez sur ce bouton pour exécuter le programme. Normalement, le robot doit avancer juste de la bonne longueur, tout en dessinant un trait rouge au fur et à mesure de son avance.

Et voilà, le premier exercice est réussi et terminé.

3 Menu principal

3.1 Programmation

Cette partie de CeeBot contient tous les exercices de programmation. Différents chapitres présentent des sujets de plus en plus compliqués. Vous pouvez effectuer n'importe quel exercice, bien qu'il soit conseillé de commencer par les plus simples.

3.2 Défis

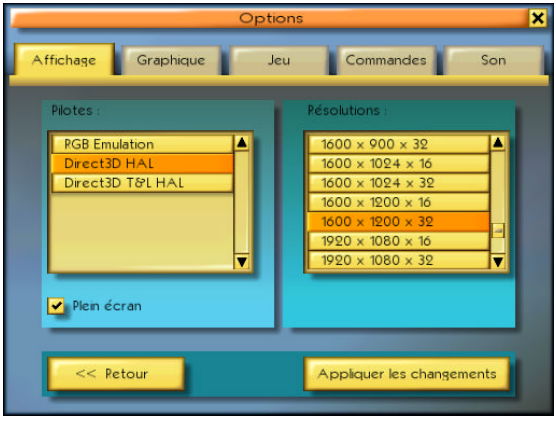
Les défis vous demandent une bonne connaissance de la programmation. Ils permettent de vérifier si les notions apprises sous « Programmation » sont effectivement assimilées.

3.3 Options

Le bouton « Options » accède aux réglages de CeeBot. Cinq onglets permettent d'atteindre les différentes catégories.

3.3.1 Affichage

Après l'installation, CeeBot utilise la résolution 640 x 480 x 16. Sur beaucoup d'ordinateurs, il est possible d'augmenter cette résolution, ce qui améliorera grandement la qualité graphique.



Pilotes :

Il faut choisir un pilote qui porte la mention HAL (Hardware Abstraction Layer). Evitez les pilotes ayant la mention « Emulation » ou « T&L ». Il arrive fréquemment que des pilotes avec des noms différents aient des comportements identiques.

Résolutions :

Les premier et deuxième chiffres indiquent le nombre de pixels en largeur et en hauteur dans l'écran. Le troisième chiffre indique le nombre de couleurs affichables (16 pour 65'000 couleurs et 32 pour 4 millions de couleurs).

Plus la résolution est grande et plus le jeu est beau. Mais il risque de devenir lent. Commencez par mettre 640 x 480 x 16. La plupart des cartes graphiques modernes supportent 1024 x 768 x 16. A vous d'essayer le meilleur compromis.

✍ Plein écran

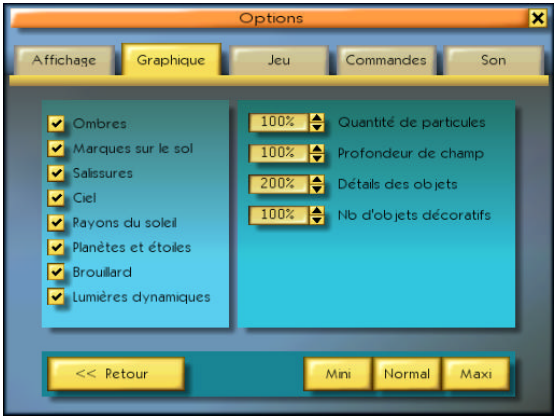
Normalement, CeeBot occupe tout l'écran, quelle que soit la résolution. Si vous enlevez la coche, CeeBot tournera dans une fenêtre fixe ayant approximativement 640 x 480 pixels.

[Appliquer les changements]

Il faut cliquer ce bouton pour que les changements effectués dans cette page prennent effet.

3.3.2 Graphique

Si le jeu est saccadé, il faut essayer de supprimer certaines options graphiques jusqu'à ce que l'affichage soit fluide. A l'inverse, si tout est parfaitement fluide, essayez d'augmenter la profondeur de champ à 200%.



Quantité de particules (0% à 200%)

Les particules servent à simuler la poussière, la fumée, les éclats, les bulles sous l'eau, etc.

Profondeur de champ (50% à 200%)

La profondeur de champ détermine jusqu'où porte votre regard. Cette profondeur est de toute façon très différente selon l'atmosphère de la planète. Une grande valeur (par exemple 200%) permet de voir loin, mais nécessite une bonne carte graphique 3D.

Détails des objets (0% à 200%)

Lorsqu'un objet est au loin, il est représenté avec moins de détails. Une grande valeur éloigne la distance à laquelle le changement est effectué.

Nombre d'objets décoratifs (0% à 100%)

Ce nombre détermine la quantité d'objets décoratifs présents, tels que les plantes, les arbres, les cristaux, etc.

4 L’écran

Pendant un exercice, l’écran ressemble à ceci :

1 Le résumé des robots ou des bâtiments existants
2 Le tableau de bord de l’objet sélectionné
3 La mini-carte
4 L’accès au menu
5 Les messages temporaires

4.1 Le résumé

La partie supérieure de l’écran résume les robots ou les bâtiments disponibles. L’objet actuellement sélectionné apparaît en orange. Lorsqu’un robot exécute un programme, le cadre clignote.

Vous pouvez cliquer sur un objet quelconque pour le sélectionner, ou appuyer sur la touche « Tab » pour sélectionner l’objet de droite.

Au départ, on trouve généralement à droite de ce bouton le cosmonaute puis le ou les robots. Un premier clic sur ce bouton montrera le résumé des bâtiments, bornes d’informations, etc. Un nouveau clic reviendra au cosmonaute et aux robots.

4.2 Le tableau de bord de l'objet sélectionné

La partie inférieure de l'écran présente tous les boutons concernant l'objet sélectionné. Ces boutons seront donc différents lorsque le cosmonaute, un robot ou une borne d'information sont sélectionnés.

4.2.1 Les 10 programmes

Pour les robots, la partie gauche donne accès aux 10 programmes. Les 4 premières lignes donnent les noms des 4 premiers programmes. Le quatrième programme contient généralement la **solution** de l'exercice. L'ascenseur permet d'accéder aux 6 programmes suivants.

Ce bouton exécute ou stoppe le programme sélectionné.

Ce bouton édite le programme sélectionné (voir chapitre 6.1). Pendant ce temps, le jeu est en pause. Vous pouvez donc tranquillement réfléchir et modifier le programme.

4.2.2 Le *Sat*Com

Le ***Sat*Com** est porté au poignet du cosmonaute. C'est avec cet appareil que vous obtenez toutes les informations sur l'exercice en cours (voir paragraphe 5).

Le premier bouton **H** active le ***Sat*Com** à la page des objectifs de l'exercice. La touche « **F1** » a le même effet.

Le deuxième bouton **[ ]** active le ***Sat*Com** sur une page qui explique le fonctionnement de l'objet sélectionné.

4.2.3 La caméra

Ce bouton permet de changer le point de vue de la caméra. Généralement, on passe alternativement d'une vue interne à une vue arrière. La touche « barre d'espace » a le même effet.

4.2.4 La réinitialisation

Ce bouton remet l'exercice dans l'état initial. Très utile lorsqu'un programme n'a pas eu l'effet souhaité, pour remettre les robots et les objets dans leurs positions de départ.

4.2.5 Les jauges

Les jauges permettent de prendre connaissance, en un clin d'œil, de l'état d'un robot. La jauge de droite n'existe qu'avec les robots volants.

4.3 La mini-carte

La partie inférieure droite est une mini-carte, qui résume la situation avec une vue aérienne. Les zones sombres correspondent à des creux, vallées, etc. Les zones claires correspondent à des points élevés, montagnes, sommets, etc.

Les robots sont représentés en jaune, les bâtiments en bleu et les ennemis en vert.

Lorsque la souris survole la carte, le symbole de l'objet visé est agrandi et une bulle d'aide en indique le nom. Un clic sélectionne alors immédiatement l'objet, même s'il est situé très loin.

Le curseur à gauche de la carte permet de choisir le facteur de zoom. Lorsqu'il est en bas, la carte montre l'ensemble du terrain. Lorsque qu'il est en haut, un grossissement x16 est effectué.

4.4 Le menu

La petite croix tout en haut à droite affiche le menu suivant :

Continuer	Ferme le menu et continue l'exercice.
Options	Accès aux réglages du jeu. Certaines options ne sont pas accessibles. Pour avoir accès à toutes les options, il faut quitter l'exercice en cours, et choisir « Options » depuis le menu principal (voir chapitre 3.3).
Recommencer	Annule l'exercice en cours pour le recommencer depuis le début.
Abandonner	Annule l'exercice en cours et retourne à l'écran des exercices.

5 Le *Sat*Com

Le ***Sat*Com** est un petit appareil bien pratique, qui affiche toutes les instructions sur l'exercice en cours, et qui contient aussi la base de données sur le langage de programmation CeeBot.

Il fonctionne un peu comme un navigateur Internet. Chaque fois qu'un mot est souligné en bleu, vous pouvez le cliquer pour afficher la page qui traite de ce sujet.

Affiche la page précédente.

Affiche la page suivante.

Affiche la page initiale, qui était visible lorsque le ***Sat*Com** a été ouvert.

Copie tous les caractères sélectionnés dans le bloc-note. Il est ainsi possible de copier un programme donné en exemple, pour ensuite le coller dans l'éditeur de programme (voir paragraphe 6.1.4).

Affiche les instructions sur l'exercice en cours. La touche « F1 » ouvre toujours directement cette page.

Affiche l'aide générale sur le langage de programmation CeeBot. C'est une véritable mine d'or, qui explique en détail la syntaxe de chaque instruction. La touche « F2 » ouvre toujours directement cette page.

Eteint le ***Sat*Com**. Sa fenêtre est donc fermée.

Lorsque l'un de ces boutons est gris, cela signifie qu'il n'a aucun effet dans la situation actuelle.

6 Programmation

Tous les robots sont programmables. Dans certains exercices, il existe un deuxième robot que vous ne pouvez pas sélectionner. C'est le cas par exemple dans l'exercice « **4.4: Le petit poucet** ». Vous ne pouvez pas programmer ce robot, car il est déjà programmé pour les besoins de l'exercice.

6.1 L'éditeur de programmes

Pour entrer dans l'éditeur de programme, il faut :

?? Sélectionner un robot.
?? Choisir le programme à éditer parmi les 10 à disposition.
?? Cliquer le bouton **{..}** « édite le programme sélectionné ».

Le jeu est automatiquement mis en pause pendant l'édition.
Si vous approchez la souris des bords gauche ou droite de l'écran, le point de vue tourne. Si vous approchez la souris des bords supérieur ou inférieur, le point de vue avance ou recule. Cela vous permet d'observer tranquillement la situation. Il est très utile de réduire la fenêtre pendant ce temps.

Taille réduite, pour réduire la fenêtre à une barre tout en bas de l'écran.

Taille maximale, pour agrandir la fenêtre sur tout l'écran.

Fermer, équivalent au bouton « D'accord ».

Si vous entrez dans l'éditeur pendant que le programme s'exécute, le jeu n'est pas mis en pause. Cela permet alors d'observer le déroulement du programme (voir chapitre 6.1.11).

La fenêtre de l'éditeur peut être déplacée en tirant la barre de titre avec la souris. Elle peut également être redimensionnée en tirant les bords ou les coins. La prochaine fois que vous ferez appel à l'éditeur, la fenêtre sera à la même position et aura les mêmes dimensions.

Les mots clés du langage apparaissent en couleur, pour faciliter la compréhension :

Couleur	***Nature***	***Exemple***
Orange	Instruction	`aim`, `fire`, `turn`, `goto`, `while`, etc.
Vert	Type d'une variable	`object`, `float`, `int`, etc.
Rouge	Constante, catégorie	`TitaniumOre`, `AlienAnt`, etc.

Lorsque le curseur est sur un mot clé du langage, la barre sous la fenêtre d'édition résume succinctement la syntaxe. Un clic sur cette barre appelle le ***Sat*Com** pour vous donner plus de précisions. La touche « F3 » a le même effet.

Un double-clic sur un mot le sélectionne rapidement. Si le bouton de la souris est maintenu pressé, les caractères sont mis en évidence. « Shift+flèche » met également en évidence quelques caractères. Les touches flèches actionnées avec la touche « Ctrl » permettent de se déplacer par mots. « Ctrl+Shitt+flèche » met en évidence par mots entiers.

6.1.1 Nouveau

Efface tout le programme en cours d'édition, et crée un squelette vide de la forme :

```
extern void object::Nouveau( )
{

}
```

`Nouveau` est le nom du programme, que vous pouvez remplacer par un nom de votre invention, mais sans espaces. Par exemple `Cherche`, `ApporteTitanium`, `RevientBase`, etc.

Remarque : Si vous avez cliqué ce bouton par erreur, il suffit d'annuler l'action.

6.1.2 Ouvrir et Enregistrer

Tous les programmes que vous créez sont automatiquement sauvegardés dans l'exercice en cours. En revanche, si vous désirez réutiliser un programme dans un autre exercice, il faut l'enregistrer explicitement, pour pouvoir le rouvrir par la suite.

Privé Le programme est mis dans un dossier privé, qui dépend donc du nom donné au début du jeu.

Public Le programme est mis dans un dossier commun à tous les joueurs.

Juste en dessous de la barre de titre du dialogue apparaît le nom du dossier. Par exemple, **Savegame\Joueur\Program** signifie que le dossier est généralement **c:\Program Files\CeeBot-A\Savegame\Joueur\Program** (selon le dossier dans lequel vous installé le jeu). Cela vous permet de récupérer un programme dans une application extérieure à CeeBot. Par exemple pour envoyer un programme par e-mail à un ami.

Les touches « Ctrl+O » et « Ctrl+S » permettent d'accéder rapidement aux fonctions *Ouvrir* et *Enregistrer*.

6.1.3 Annuler

Annule la dernière action effectuée dans le programme. Il est ainsi possible d'annuler en tout temps les 20 dernières modifications.

Le touche « Ctrl+Z » a le même effet.

6.1.4 Couper, Copier et Coller

Coupe ou *copie* les caractères mis en évidence, pour pouvoir les *coller* ailleurs. Si aucun caractère n'est mis en évidence, c'est toute la ligne qui est prise en compte.

Les touches « Ctrl+X », « Ctrl+C » et « Ctrl+V » permettent d'accéder rapidement aux fonctions *Couper*, *Copier* et *Coller*.

6.1.5 Taille des caractères

Ce petit curseur permet de choisir la taille des caractères pendant l'édition et dans le ***Sat*Com**. Cette taille est mémorisée jusqu'à ce qu'elle soit changée à nouveau.

6.1.6 SatCom : Instructions

Donne les objectifs de l'exercice. Vous pouvez aussi appuyer sur « F1 ».

6.1.7 SatCom : Aide à la programmation

Donne les instructions générales sur la programmation, à partir desquelles vous avez accès à toutes les explications spécifiques par des liens hypertextes. Vous pouvez aussi appuyer sur « F2 ».

6.1.8 D'accord

Compile le programme puis quitte l'éditeur si aucune erreur n'est détectée. En cas d'erreur, la partie incriminée est mise en évidence et la barre sous la fenêtre d'édition indique la nature de l'erreur.

6.1.9 Annuler

Quitte l'éditeur sans compiler le programme. Toutes les modifications effectuées dans le texte sont néanmoins enregistrées.

6.1.10 Compiler

Compile le programme, sans quitter l'éditeur. Cela vous permet de vérifier si le programme contient ou non des erreurs.

6.1.11 Démarrer/stopper

Démarre ou stoppe le programme, sans quitter l'éditeur. Cela permet de faire du *debug*, c'est-à-dire de chercher les causes d'un éventuel mauvais fonctionnement. Lorsque le bouton de droite représente un homme avec les mains sur les hanches, le programme est exécuté en *pas à pas*, c'est-à-dire instruction par instruction.

Remarque : Lorsque le programme est en exécution, la fenêtre d'édition devient orange. Vous pouvez alors observer le déroulement du programme, mais vous ne pouvez plus le modifier.

6.1.12 Pause/continuer

Passe alternativement du mode *pas à pas* au mode sans interruption.

6.1.13 Un pas

Exécute l'instruction suivante, en mode *pas à pas*. La partie sous la fenêtre d'édition montre alors le contenu des différentes variables du programme. Ce contenu change à chaque pas, en fonction des affectations effectuées.

6.2 Le langage CeeBot

Le langage CeeBot est très proche de Java？, de C++ et de C#. Il est adapté à une approche pédagogique. Dans ce manuel, nous ne présentons que quelques exemples simples. Pour avoir une description complète, utilisez le ***Sat*Com** en appuyant sur « F2 ».

Attention de respecter les majuscules et les minuscules. Par exemple, si vous écrivez `Radar`, le programme ne sera pas compilé. Il faut écrire `radar`.

Chaque instruction est terminée par un point-virgule `;`.

Un commentaire commence par `//` et se termine à la fin de la ligne. Il n'a aucune incidence sur le déroulement du programme, mais sert uniquement à le rendre plus explicite.

6.2.1 L'instruction `radar`

Avec l'instruction `radar();`, le robot peut chercher des objets comme des ennemis, des robots ou des bâtiments. Donnez entre parenthèses le nom de l'objet que vous cherchez. Mettez le résultat dans une variable de type `object`. Voici par exemple comment chercher la fourmi la plus proche :

```
// Tout au début du programme:
object  chose;  // déclaration de la variable

// Cherche la fourmi la plus proche:
chose = radar(AlienAnt);
```

6.2.2 L'instruction `goto`

L'instruction `goto();` permet de déplacer le robot vers une certaine position. En général, on déplace le robot vers un objet qu'on a détecté avec l'instruction `radar();`. Si on a mis les informations rendues par `radar();` dans une certaine variable, il faut écrire le nom de la variable suivi de `.position` pour obtenir la position de l'objet. Voici par exemple comment chercher un cube de titanium, aller vers la position du cube et le saisir :

```
object chose;
chose = radar(Titanium);
goto(chose.position);
grab();
```

6.2.3 Les instructions `grab` et `drop`

L'instruction `grab();` permet de saisir avec la pince du bras manipulateur un objet qui se trouve au sol, sur l'emplacement arrière d'un autre robot ou sur le socle d'un bâtiment. L'instruction `drop();` dépose l'objet actuellement transporté. Voici par exemple comment saisir ce qui se trouve devant le robot et le poser 5 mètres plus loin :

```
grab();   // prend l'objet
move(5);  // avance de 5m
drop();   // dépose l'objet
```

6.2.4 L'instruction `fire`

L'instruction `fire();` permet de faire feu avec le canon du robot. En général, on tire des rafales d'une seconde :

```
fire(1);
```

6.2.5 L'instruction `while`

L'instruction `while() {}` permet de répéter des instructions plusieurs fois. L'utilisation la plus courante de `while` consiste à répéter des instructions encore et encore, à l'infini. Pour ce faire, on écrit `while (true) {}` et on met les instructions à répéter entre les accolades `{}`. Voici comment répéter le fait de chercher une araignée, de se tourner vers elle et de tirer :

```
while (true)
{
	chose = radar(AlienSpider);
	turn(direction(chose.position));
	fire(1);
}
```

Il suffit d'exécuter ce programme une seule fois, il tuera toutes les araignées autour de lui.

6.2.6 L'instruction `distance`

Avec l'instruction `distance( , )` vous pouvez calculer la distance entre deux positions. Si vous écrivez `position` tout seul, cela donne la position du robot qui exécute le programme. Si vous écrivez le nom d'une variable suivie de `.position`, cela donne la position de l'objet décrit par la variable. Voici comment avancer d'une distance égale à la distance entre le robot et la fourmi la plus proche :

```
chose = radar(AlienAnt);
move(distance(position, chose.position));
```

Ceci est bien sûr parfaitement suicidaire, mieux vaut s'arrêter 40 mètres avant, pour être à la bonne distance de tir :

```
chose = radar(AlienAnt);
move(distance(position, chose.position) - 40);
```

6.2.7 L'instruction `if`

L'instruction `if() {}` permet d'exécuter des instructions seulement à une certaine condition. La condition est donnée entre parenthèses `()`, les instructions entre accolades `{}`. Voici par exemple comment faire en sorte que le robot tire seulement si la distance à la cible est inférieure à 40 mètres :

```
chose = radar(AlienAnt);
if (distance(position, chose.position) < 40)
{
	fire(1);
}
```

Si l'instruction `radar ();` ne trouve pas l'objet cherché, elle rend la valeur `null`. Ainsi il est possible de tester si un objet n'existe pas avec `(chose == null)`, ou de tester si

l'objet existe avec `(chose != null)`. Deux signes égal == testent l'égalité, un point d'exclamation suivi d'un signe égal != teste l'inégalité. Voici comment aller se recharger seulement s'il y a une station de recharge :

```
chose = radar(PowerStation);
if (chose != null)
{
        goto(chose.position);
        wait(5);
}
```

6.2.8 L'instruction `motor`

L'instruction `motor( , );` permet de donner directement une vitesse aux moteurs gauche et droite du robot. `motor` conserve la vitesse du moteur pendant l'exécution des instructions suivantes. Ceci est utile pour faire tourner le robot pendant l'instruction `fire();`. Ainsi, il est possible de balayer toute une zone avec la rafale. Voici par exemple comment balayer toute la zone qui se trouve devant le robot :

```
turn(45);          // tourne à gauche de 45 degrés
motor(0.5, -0.5); // rotation lente à droite
fire(2);           // feu
motor(0,0);        // immobilise le robot
```

En faisant avancer le moteur gauche à demi-vitesse et en faisant reculer le moteur de droite à demi-vitesse, le robot tourne lentement sur lui-même pendant la rafale de 2 secondes.

6.2.9 L'instruction `turn`

Utilisez l'instruction `turn();` pour faire tourner le robot d'un certain nombre de degrés sur lui-même. 90 degrés signifie un quart de tour à gauche, 180 degrés signifie un demi-tour, 360 degrés signifie un tour complet. Voici quelques exemples avec `turn();` :

```
turn(90);   // quart de tour à gauche
turn(-90);  // quart de tour à droite (négatif)
turn(180);  // demi-tour
```

Pour se tourner vers un objet trouvé avec l'instruction `radar();`, il faut calculer l'angle de la rotation avec l'instruction `direction();` :

```
chose = radar(AlienSpider);
turn(direction(chose.position));
```

Puis il suffit de faire feu, et il y a un élément hostile en moins.

7 Développeur

Daniel Roux, Denis Dumoulin, Otto Kölbl, Michael Walz

EPSITEC SA, Mouette 5, CH-1092 Belmont
ceebot@epsitec.ch, www.epsitec.ch