Quelques pensées sur le projet Lego

1. **Choix de logiciel/langage**. Il y a quatre options principales :
   a. Lejos avec NXJ (http://lejos.sourceforge.net) - une version de Java pour NXT. Les choses qui manquent est le capteur de distance à ultrason, le vieux capteur de rotation (c'est compensé par un bon support du nouveau capteur intégré au moteur, et surtout Bluetooth (il faut donc charger le programme via USB). Par ailleurs, je n'arrive pas à télécharger,
   b. Microsoft robotics studio http://msdn.microsoft.com/robotics Vous pouvez regarder, c'est peut-être un peu lourd
   c. Ou bien BricxCC http://bricxcc.sourceforge.net/ avec NBC/NXC qui donne un support assez complet de lego NXT. NBC est un assembleur, NXC une version de C. Ce logiciel me semble un poil plus mur que Lejos.
   d. Finalement, si vous choisissez le vieux lego vert (RCX) vous aurez l'embarras de choix.
2. **Modes de moteur.** Le moteur est couplé avec un capteur de rotation. Le plus intéressant qu'il y a déjà un contrôleur feed-back configurable, qui peut fonctionner en plusieurs régimes (voir par exemple le chapitre VIII du tutorial NXC). Les modes SPEED et IDLE sont particulièrement intéressants pour vos projets.
   a. SPEED MODE, ou grace à un contrôle par feed-back, le moteur tourne à la vitesse souhaitée.
   Par exemple en faisant
   ```
   OnFwdReg(OUT_AC,50,OUT_REGMODE_SPEED); en NXC
   ou bien
   Motor.A.regulateSpeed(true);
   Motor.A.setSpeed(500) : en NXJ
   ```
   vous aurez le moteur à 50% de vitesse indépendamment de la charge. C'est un régime très intéressant, si vous l' utilisez pour déplacer un véhicule, sa vitesse sera proportionnelle au signal d' entrée, ce qui correspond à un modèle de 1er ordre : x'= ku. (avec x la position, u l'entrée, et k un coefficient facile à mésurer). Pour un « bras » l' équqtion serait la même : x'= ku avec x l'angle.
   b. Le contrôle PID `RotateMotorPID(OUT_A, 100, 180, P, I, D);`
   Je ne crois pas que c' est un bon choix, parce que vous devrez analyser ce controleur PID et le votre. Trop complexe.
   c. Le régime simple IDLE, où le moteur est commandé directement par l'entrée.
   Par exemple en faisant
   ```
   OnFwdReg(OUT_AC,50,OUT_REGMODE_IDLE); en NXC
   ou bien
   Motor.A.regulateSpeed(false);
   Motor.A.setSpeed(500) : en NXJ
   ```
   Pour ce régime la puissance du moteur dépend directement du signal d'entrée (l' argument de la fonction). Il n'y a pas de régulateur feedback caché. Donc la vitesse de rotation est loin d'etre constante pour une entrée

donnée. On peu espérer que le moment de force (torque) développé par le moteur est proportionnel (à peu près) au signal d'entrée. Si vous utilisez un tel régime dans un véhicule, l'équation sur la coordonnée serait x'' = ku. Meme chose pour l'angle d'un bras qui tourne dans le plan vertical, mais il faut tenir compte du moment variable de la force de gravitation :
x''=k1*u+k2*sin x. Pour les petites oscillations on linéarise cette équation en remplaçant sin x par x (voir un livre de mécanique).

d. Par ailleurs il y a des possibilités de synchroniser deux moteurs, mettre un moteur à une position donnée etc

3. Capteurs de rotation

a. Intégré au moteur :
Pour les mettre à 0
```
ResetTachoCount(OUT_A) en NXC
ou bien
resetTachoCount(); // en NXJ
```

```
Pour les lire
x = MotorTachoCount(OUT_A); en NXC
ou bien
int angle = Motor.A.getTachoCount(); // en NXJ
```

Il existe une différence que je n' ai pas comprise entre TachoCount et Rotation Count

b. L'ancien capteur RCX utilisé via un adaptateur. Non supporté en NXJ/LEJOS. Pour NXC voir tutorial, page 36.

4. Que faire ?

- Un projet simple serait un petit véhicule avec le moteur en régime SPEED pour lequel on contrôle sa position par arapport à un obstacle
- Un projet très ambitieux serait le meme véhicule qui balance un pendule inverse (grace au régime SPEED qui permet de controler directement la vitesse, ca devient un peu plus simple)
- Entre les deux il y a toutes sortes des bras et pendules.
- On peut aussi controler la direction du movement d' un véhicule. Soit avec une roue de direction que vous positionnez avec RotateMotor, soit avec les vitesse de la roue gauche et le roue droite

5. Signal de référence En principe vous pouvez donner à votre système n'importe quel signal de référence. Mais pensez comment faire la démonstration amusante. Par exemple, permettez à l'utilisateur de commander le système par un joystick. Ou donnez une sinusoïde ou autre square wave pour faire votre système se déplacer périodiquement. Ou donnez un signal en rythme de musique, et faites votre robot danser en jouant un tube.

Bon courage,
EA