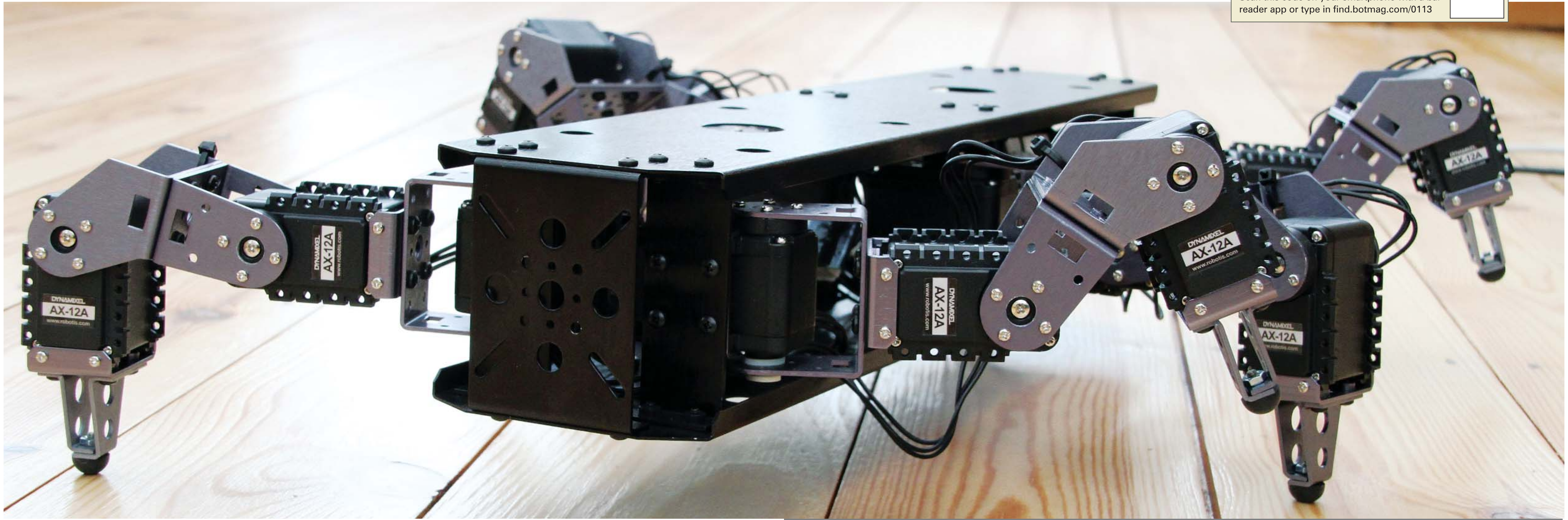
by Carl Owen

SEE THE VIDEO!
Scan this code on your smartphone with a bar reader app or type in find.botmag.com/0113



PROGRAMMING A HEXAPOD WITH KINEMATICS

Welcome to the FlowStone workshop number 8, where we give a beginner's guide to computer programming using the FlowStone free graphical programming language. In this issue we are going to look at programming a robotic hexapod using Inverse Kinematics (IK).

WHAT IS A HEXAPOD?

You'd be hard pushed to read this magazine for a few issues without coming across a hexapod or two; basically, these are robots with six legs (In Greek Hex means six and Pod means feet). Various companies make robotic hexapods and today we are using the all metal Smart Hexapod from CrustCrawler.

WHAT IS IK?

IK or Inverse Kinematics is the Math behind almost all robotic movement in the real world. To be honest it is vastly complicated and comes in many forms but fortunately you don't need to totally understand it to use it. Thankfully, in FlowStone you get an IK module pre-made for you to use.

To explain IK in simple terms it is the calculation of the angles required for each of a robot arm's joints in order for it to move to a specific position in 2D space in a given period of time. As humans we don't use Inverse Kinematics but instead use Forward Kinematics, which is the constant iteration of movement until you get to where you want to go; the main difference is that humans can see in 3D and use touch and feel and plus we learn over many years. Technically it would be possible to use Forward Kinematics for Robotics but would greatly add to the cost and complexity.

THE BUILD

Before we dive into code it's worth saying something about the CrustCrawler hexapod hardware. This arrived in a large box with an excellent printed manual full of photos and diagrams and was a joy to build. Everything fitted together perfectly and because there was a lot of repetition, since there were six identical legs, it went together surprisingly quickly. In this kit there are 18 Dynamixel AX-12 Robotic servos that are intelligent, so they switch off if the torque exceeds a set limit, saving the servos from burning out. This will save you a fortune in the long run. Each servo has a unique ID which is soft programmable so we programmed ours from 1 to 18 left to right, front to back. We made a simple FlowStone application to do this that allows you to program the servo ID and test it straight away (available as a download from the DSPRobotics Site). Also the Dynamixel servos only use three wires with servos daisy chained throughout the whole system so the wiring is so much simpler than with traditional RC type servos that have three wires each (i.e., 3 vs. 54 Wires!).

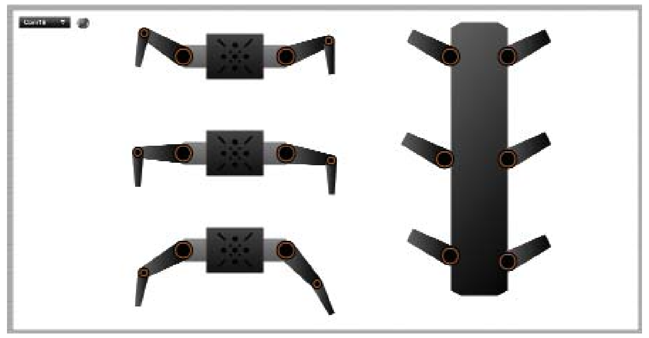
Robotic First Steps

When looking at taking the first steps into robotics there are many technical issues you need to consider before choosing your hardware, be it a bi-ped, humanoid, robot arm or hexapod. Having now programmed all of these to some level it is clear to me that there are only two ways to go for the beginner, robot Arm or hexapod. The reason I say this is twofold, firstly as a rule they are stable and don't fall over all the time, and secondly, it is the wide range of functions you can create from a programming point of view with relative ease. Robot arms are great as a first step as you can program them to move using IK and then pick things up or systematically move levers, etc. These are great for education. Hexapods are basically six robot arms on one body and can also be programmed with IK, but they offer so much more to keep you interested in the future, e.g., organic movement to mimic nature, and various walking Gaits to simulate different insects and creatures. Plus they walk without falling down on uneven surfaces, etc. So, as you can probably, tell I'm a total hexapod convert!

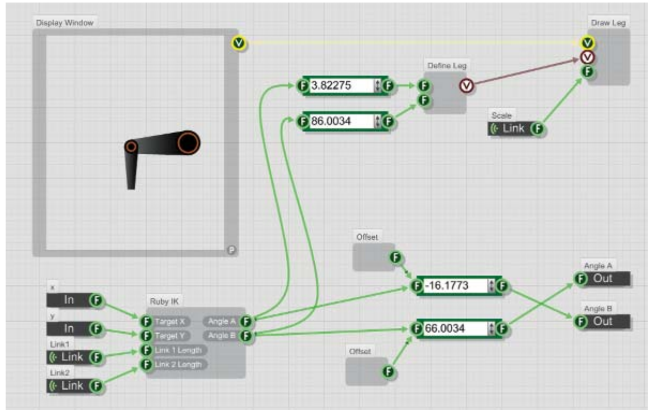
THE PROGRAMMING

One thing we have learned when programming robots is that before you get it right you will get it wrong and your robot will curl up and try to kill itself! We saw this first when we made the educational

Flowbotics Studio robot arm software for Project Lead the Way (PLTW.org). It is so much easier to program robotics especially with IK when you have a graphical simulation on screen FIRST to get the basics working before you even connect the hardware. So we decided that due to the complexity of a hexapod that it was worth the effort to make an onscreen representation of our hexapod to speed up the development time.


Hexapod Onscreen Simulation.

To do this we used some of the powerful Ruby Code programming tools inside FlowStone to draw the legs and body and then manipulate them using angles for each joint. Note the front view of each set of legs and the top view of the body joints.
In order to program each leg we treated them like individual robot arms, six of them. Each arm would need its own IK module to tell it where to go in 2D space.
To do this you need to give the IK module four pieces of information: the length of the two arm segments and the X and Y coordinates of the position you would like the arm to go to. Providing these are all in range the IK module spits out the two angles required for each joint in order to move to that location. So, quite quickly you can have a leg moving on screen by just feeding it with X and Y co-ordinates.

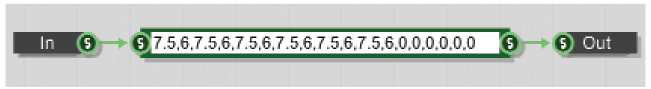


A single Leg with the IK Module.

The rotation joint (as seen from the Top View) was just a single joint and therefore didn't require IK as it was just a single angle.

Once this was working it was just a case of duplicating this six times to make our complete hexapod simulator. To control our six legged beast we used an Array of data comprising each of the six leg positions (supplied as X,Y coordinates in cm) and six rotational shoulder joint angles (in degrees).

So a typical array looks like the representation in the illustration captioned "Array of leg positions and angles."



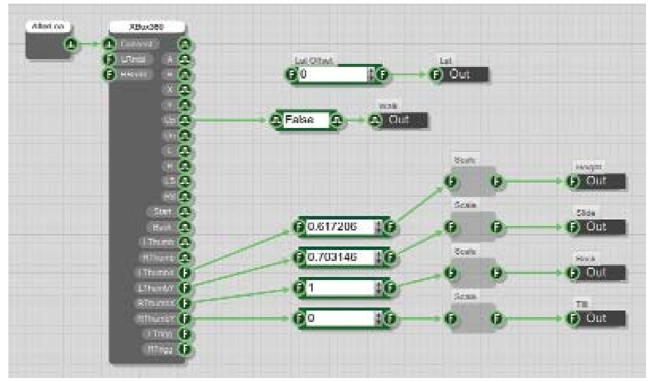
Array of leg positions and angles.

In fact this is our 'home' position with all of the joints at 90°. We could now manually type these values in and see our simulation move.

XBOX CONTROL

USB Xbox Controller.

To save continually typing in co-ordinates we thought it would be fun to connect an Xbox USB games controller to the software to control the movement! Using the built-in Xbox module in FlowStone we took some of the thumb controls and used them to control our hexapod. The only conversion necessary was to scale the Xbox inputs that are +/- 1 to match our hexapod ranges.



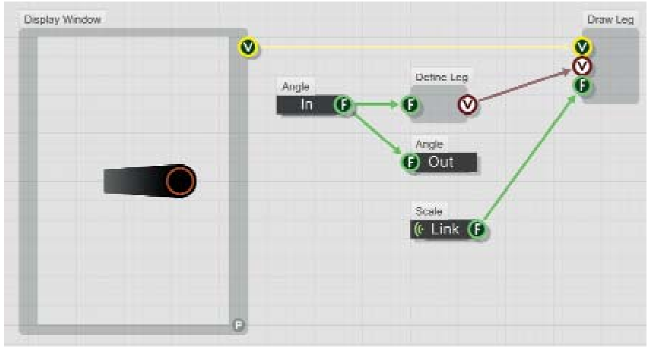
Xbox controls in FlowStone.

These values were then added or subtracted from the Array coordinates to make our hexapod move while standing still.

CONTROLLING THE REAL HEXAPOD

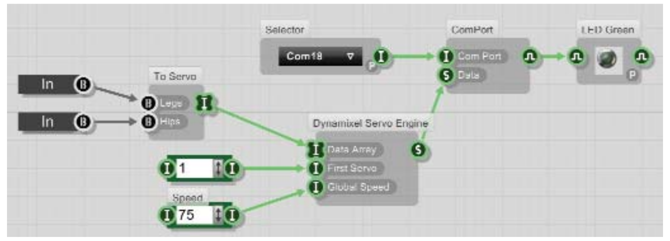
Hexapod, Laptop & Xbox Controller.

Next was the real exciting part; connecting our simulation to the real hexapod hardware! Since we are controlling our hexapod via the PC we used the 'USB2Dynamixel' convertor to connect to the computer. This gives you a virtual COM port that can be easily programmed in FlowStone to communicate with your servos. Since we



Leg rotation joint with a simple angle input.

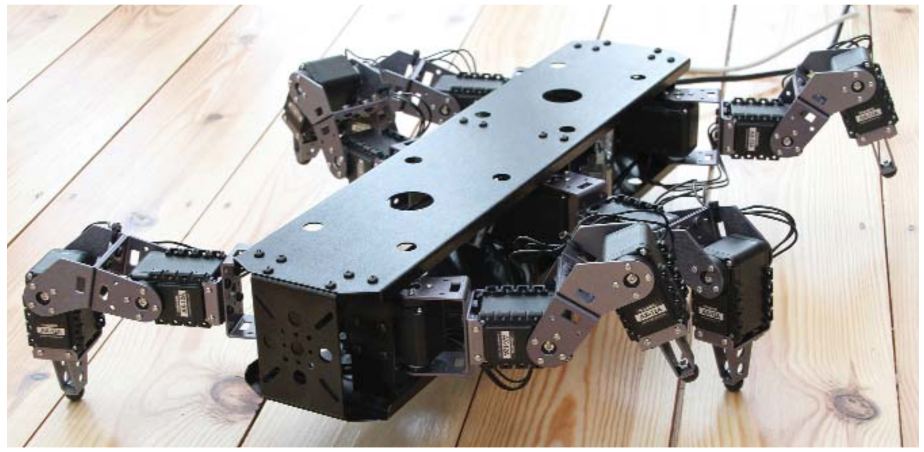
had already programmed the CrustCrawler Smart Robotic Arms last issue we already had our Dynamixel Servo Module made to convert the arrays of servo positions into the HEX (Hexadecimal) strings of data to communicate with the servos. We used a special command (0xFE) that allows you to send the entire control data for all of the servos and then co-ordinate a single move command so that all of the servos move at exactly the same time.



Dynamixel Servo module & Comport.

We connected it to the power and the hexapod jumped into life! There were a few offset errors where the angles were different from our simulation due to some of the 90° offsets required in the build but these we soon corrected for by adding some constants to the angle data. Wow it works! We could now adjust the basics of height and lateral forward and backwards movements with all of the legs planted firmly on the floor. Next we wanted to add some more organic movements to bring our creature to life. What was surprising was that because we were using IK these we're really simple!

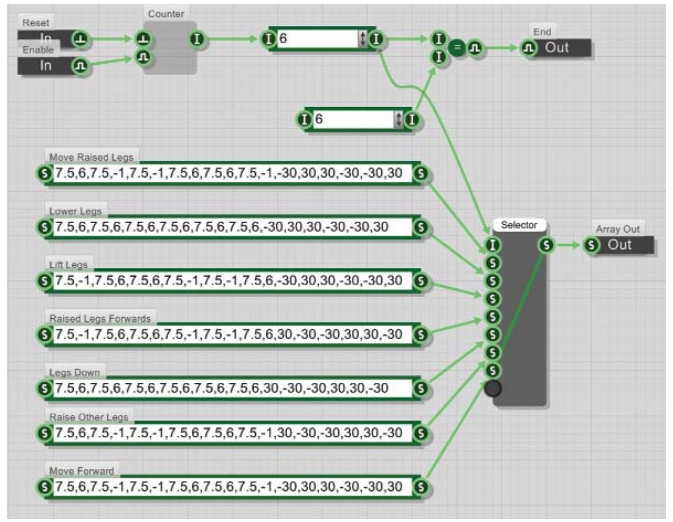
For example to make our hexapod bow down or rock side to side we just added a +ve and -ve offset to the opposing legs. Even better using our Xbox controller these could be done simultaneously making our hexapod move around in a scarily real life looking motion (See the Video).


Our hexapod: 'Lizzard Gait'.

HEXAPOD GAIT

Next was to try and make our hexapod walk, this is a little more complicated as first we needed to decide on a gait. hexapod gaits are the different ways it can walk, with different walking styles representing different gaits. Having six legs there are hundreds of possibilities that you can program to make it walk in different ways. For example you could just move one leg at a time or three legs at a time. We decided on what I call the "Salamander Gait," which emulates a giant lizard. This gait requires lifting three legs at a time (front and back on one side and middle on the other) creating a stable tripod base. The lifted legs would then move forward in the air, then return down to the ground before the other three legs were lifted allowing the robot to move forwards on three legs at a time.

To program this we made three different sets of IK Arrays, one to go from our 'Home' all square position to starting to walk, one to walk that could be repeated indefinitely, and one to return from walking back to the home position. These arrays were sent to the hexapod in timed steps to allow it time to move. Finally these were sequenced using a counter and selector and then controlled via the Xbox controller.



Move Raised Legs
7.5,6,7.5,-1,7.5,-1,7.5,6,7.5,6,7.5,-1,-30,30,30,-30,-30,30

Lower Legs
7.5,6,7.5,6,7.5,6,7.5,6,7.5,6,7.5,6,-30,30,30,-30,-30,30

Lift Legs
7.5,-1,7.5,6,7.5,6,7.5,-1,7.5,-1,7.5,6,-30,30,30,-30,-30,30

Raised Legs Forwards
7.5,-1,7.5,6,7.5,6,7.5,-1,7.5,-1,7.5,6,30,-30,-30,30,30,-30

Legs Down
7.5,6,7.5,6,7.5,6,7.5,6,7.5,6,7.5,6,30,-30,-30,30,30,-30

Raise Other Legs
7.5,6,7.5,-1,7.5,-1,7.5,6,7.5,6,7.5,-1,30,-30,-30,30,30,-30

Move Forward
7.5,6,7.5,-1,7.5,-1,7.5,6,7.5,6,7.5,-1,-30,30,30,-30,-30,30

Walk Sequence Array.

So now our hexapod can walk! The next step would be to program in a range of different gaits to walk sideways, walk in a circle, backwards, or walk like an ant or spider, etc. But for this article I think this is enough to give you the basics. You could also make your hexapod wireless by mounting a small PC (FIT-PC) on-board and using a wireless Xbox Controller, as shown on the video on the CrustCrawler site.

CONCLUSION

Programming this hexapod has been very rewarding and fun. What's really cool is that I've only scratched the surface; there is a long way to go in order to get the most out of a hexapod. I could see that for education a hexapod, albeit more expensive than a Robotic Arm, would bring far more challenges, involvement and versatility to the educational experience.
This could also engage more students due to the range of possible exercises. For example some students could investigate the different gaits of insects, lizards and spiders; teams could work on programing different gaits before bringing them all together to test, etc.

The key to all of this is using IK and visualizing it in software. As always, we will post the source code of the examples on our DSPRobotics examples page at www.dsprobotics.com for you to download and play with. Even if you don't have a hexapod you can still play with the software if you have an Xbox controller and see the simulation working.

Links
CrustCrawler, www.crustcrawler.com, (480) 577-5557
DSPRobotics, www.dsprobotcs.com
Robotis, www.robotis.com, (949) 333-3635

For more information, please see our source guide on page __.