DogTrack User Guide

Dogtrack is an adaptation of 2073's JeVois FRC vision suite using a modified vision pipeline. All due credit goes to 2073 and MrBill for the fantastic work they've done in the offseason and shared with the community to make vision more accessible to FRC teams everywhere. For those unfamiliar JeVois is 50$ open source camera/coprocessor with a variety of applications to computer vision and robotics problems. Much like EagleTrack, DogTrack is intended to provide a very easy to setup FRC targeting solution for new/inexperienced programmers. This type of solution is offered off the shelf by things like Limelight but at a higher cost. In both cases target posistion provided over serial to the Robo-Rio, but the calculation of distance/angle and other target info is left to the end user. This guide details the setup and operation of the system for new users.

Concept of Operation

Dogtrack consists of three python programs that are used to tune and operate the Jevois Camera, working with an LED ring light, to track retroreflective tape targets. **DogTuner.py** is a python 2.7 program designed to run on a desktop/laptop connected to JeVois. DogTuner sends serial commands over USB to **DogTrackCal.py** when it is running on JeVois. DogTrackCal returns a binary image with identified targets drawn on top. This allows the user to use the sliders on DogTuner to adjust the values Jevois uses to filter out its target. When you are done tuning the values are exported to a calibration file used by **DogTrack.py**. DogTrack by default runs at a video output of MJPG 320x240 at 15 FPS. It outputs data about targets it is tracking at 60 FPS. With reasonable settings and lighting this allows 60 FPS tracking while maintaining a useable camera view for the driver to reference.

Initial Setup

This section guides the user through the initial setup of JeVois on a computer running windows 10. Programming JeVois on a computer running Linux is easy and possible, but already covered thoroughly in the JeVois website. This guide helps with some of the sticking points of running JeVois in windows.

Hardware Needed:

1. JeVois Camera with latest image on SD card: If you don't have this follow the guide on the JeVois website to set it up.
2. Green USB Ring Light
3. 3D printed Camera Mount

4. USB Mini-B to A cable: One is included with the Kit that has two USB A plugs. This is for compatibility with older 1.0 USB ports that provide less power. One plug should be sufficient on most computers/the roborio.
5. 4X Button Head screw, 2-56 thread, 3/16 inch length. You can find these at McMaster or your local hardware store.
6. Retroreflective tape in the Shape of the FRC vision target you're trying to track.

Software Needed:

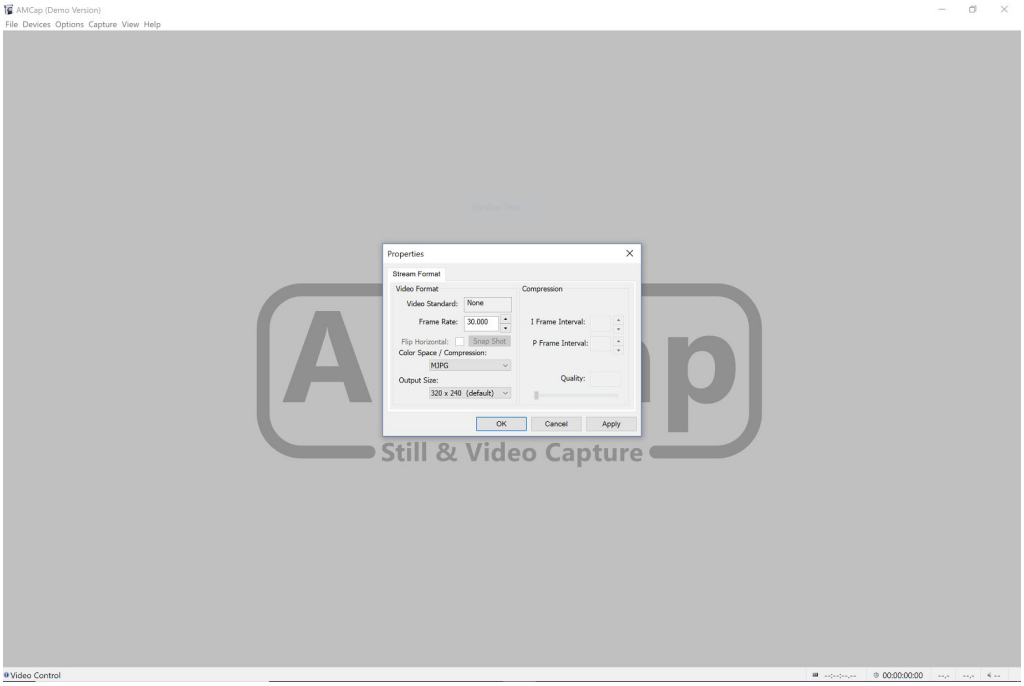
1. Arduino IDE: Used for communicating with JeVois over Serial.
2. Python 2.7: Important that it is **not** 3.6.
3. AmCap: Program to view webcam output. There are two versions online, use this one.
4. Python Serial: Go to the cmd prompt and type "pip2.7 install python-serial" Python will need to be added to the system PATH to work properly. If this doesn't work google "adding python to path." You'll also need tkinter, which should be included in python 2.7.

Initial Setup:

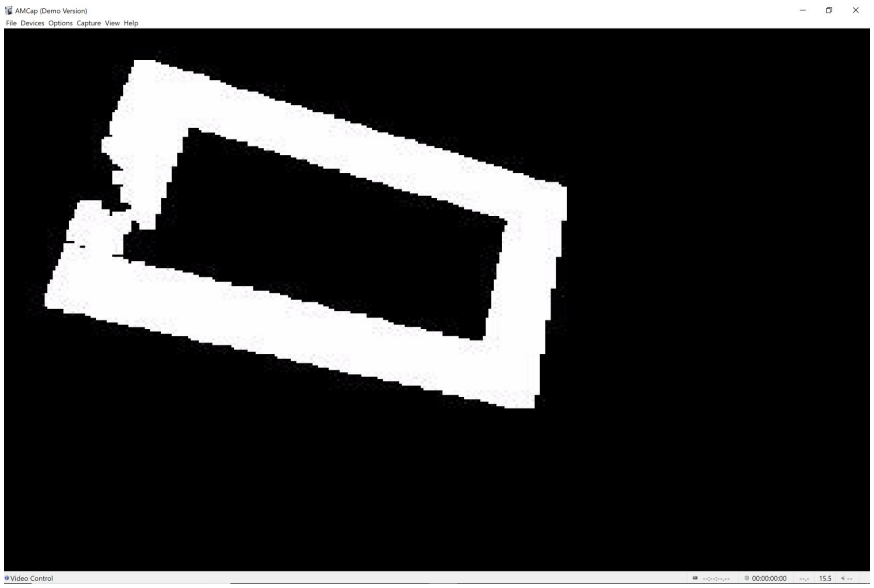
Follow these steps if DogTrack hasn't been added to the system yet. If the SD card is already loaded with DogTrack plug in the camera and skip to step 12.

1. Download the repository from Here
2. Plug the USB cable into your computer and Jevois.
3. You should see the LED light on jevois light up green, if it does not you don't have good power to the camera. Troubleshoot your usb port/cable.
4. After a few seconds an orange light should join the green one. This indicates Jevois is booted up and ready to go. WIndows should give you a "ding" to let you know it's recognized the camera.
5. Open up the Arduino IDE. Then open the Serial Monitor by pressing ctrl+shift+m. This is should open up a serial communication with Jevois.
6. Type "help" and press enter. If everything is working Jevois will send back a lot of text about its settings and options
7. Type "usb sd" and press enter. This will cause JeVois to export it's SD card as a virtual disk (like a flash drive) so you can add modify and remove files from it.
8. At this point you'll be copying over the files from the github repository to Jevois. The two folders "DogTrack" and "DogTrackCal" should be copied into "Modules/Jevois" folder on Jevois. The two files in the "Config" folder should be copied into the "Config" folder on Jevois. Replace any duplicate files. You don't need to copy over "DogTuner.py" because it only runs on your computer, Not JeVois.
9. Now that you've copied over all the files you'll eject JeVois. Do this like you would any other removable drive, from the Icon in the system Tray.
10. After doing this JeVois will automatically reboot. You'll have to close out the Serial window in the Arduino IDE every time you do this, rebooting kills the connection.
11. For the next step you'll need to close the arduino IDE altogether, this releases the serial port for DogTuner to use.

12. Open up AMCap. You'll probably see video feed from JeVois, if not select it from the "devices" menu. It should be labeled "video control" or something similar.
13. AMCap automatically selects a resolution for the JeVois to run at. JeVois runs different Modules based on which resolution and FPS your select. So you'll probably see it doing something that isn't FRC vision tracking.
14. Go to Options>Video Device> Capture format and Select MJPG 320x240 at 30 FPS. See the picture below.



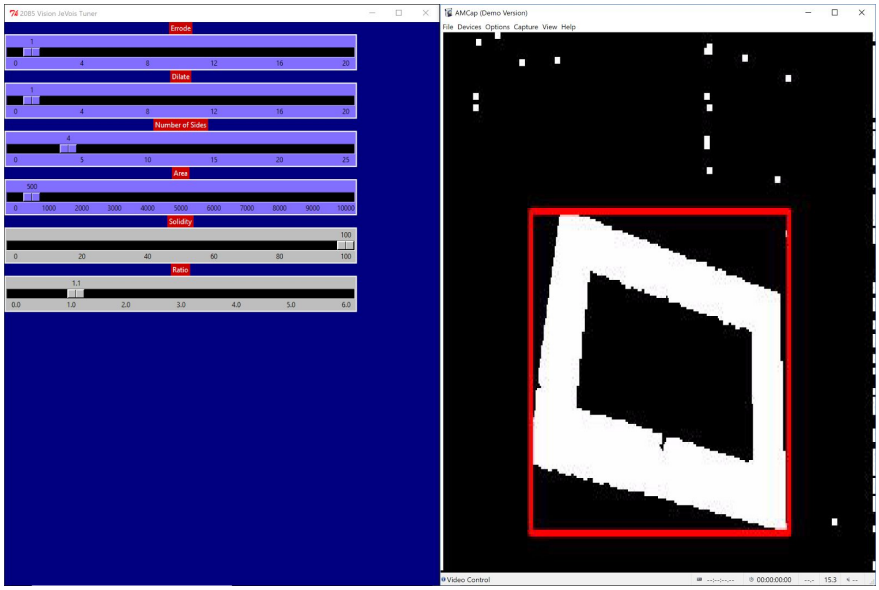
15. At this point you should see some static on a mostly black screen. This is because DogTrackTuner (the module set to load when you select MJPG 320x240@30 FPS) looks for green light that is fairly unique to green leds.
16. At this point you should attach JeVois to the 3D printed mount, zip tie on the LED, connect the LED to 12v power, and point the whole assembly at a retroreflective target. You should see something like this in in AMCap:



By default JeVois is finding all the “green” pixels and setting them to 255 (pure white), and setting everything else to 0 (Pure Black).

17. Right now DogTrackTuner is running with some default settings it has. We want to tune it to detect the particular target you’ll be using this year. To do this open “DogTuner.py”

18. You should see something like the Image below when you open DogTuner. If DogTuner fails to open you should double check that the Arduino IDE is closed and the the com port number in the DogTuner.py is correct for your system.



The sliders you see are used to tune the “vision pipeline” that Jevois is using the detect the target. It is extremely worthwhile to read some documentation on computer vision to get an idea what each of these sliders do. They are (extremely) brief overviews of each below.

Erode: Reduces the size of pixel clusters. This effectively removes noise from the image. The Slider controls the number of times the image is eroded. One is normally sufficient. More erosion cycles take more time and slow down the overall FPS the program runs at.

Dilate: Effectively the "*opposite"* of erode. It increase the size of pixel clusters. Running an erosion and dilation is a common strategy to remove the noise and then restore an image to it's normal size. Feel free to play around with this. Not using a dilate step (slider set to 0) is faster, but your clusters will be less defined.
Find Contours: You don't have a slider for this, but it is an intermediate step where the program finds all the clusters of pixels it can and outlines them. You'll see these in blue on the output. Drawing all of these bad contours on the image takes a while, which causes the calibration program to run more poorly.
Number of Sides: This algorithm filters contours by what polygon they are closest to. For a rectangular target the default setting should be fine.
Area: This is used to filter out any contours that don't meet a minimum area. Essentially we're discarding all the contours too small to be the target. Remember that the target will appear smaller depending on how far away your robot is from it. You'll have to play around with the camera at the different positions your robot will identify the target from.
Solidity: Used to filter out jagged or incomplete contours. This filter identifies how close to a rectangle the object contour is. By default this is a horizontal rectangle so a tilted target, or the camera being tilted would decrease the solidity. For most previous games the assumption that the camera and the target are level was a good one. This will not work as well for circular or triangular targets as currently implemented.
Ratio: This filters targets based on their aspect ratio (width/height). Contours that are below the set aspect ratio are discarded. This is particularly useful if there are multiple targets of different shapes. Note that the robot looking at the target from off center will change the aspect ratio.
After playing around with these sliders you should be able to get a red rectangle (indicates the camera is tracking the target) around your selected object from all the positions on the field that would be useful to you. Hollow targets can end up with two tracking rectangles on them. This is fine and is something you'll deal with on the RoboRio.
19. Now that you have found settings that work for you, you can switch over to DogTrack, which is the program you'll run for actual targeting. First close out python and dogtrack. Open up the Arduino IDE and open up the serial monitor (again ctrl+shift+m). DogTrack will freeze up if it doesn't have an open serial link to send target data to.
20. Go to the screen where you selected the resolution and change the FPS number to 15. You'll probably have to manually type this in.
21. DogTrack will automatically load up the settings you last had in DogTrackCal and use them. You should see a normal camera output with a red tracking rectangle over any targets you've been working with. You'll also see FPS numbers printed on the image. These will vary but ideally should stay above 60 to avoid dropping frames. Test this at various locations on the field to see how well it works. You'll also see target info being printed into the serial monitor. This is the same data you'll use on the Roborio to track objects. If everything is ok you can move on. If not go back to DogTrackCal.
22. JeVois is now ready to be used for FRC targeting. You'll plug the USB cable from JeVois into the RoboRio and receive target info and the video stream over that link.

Using JeVois with the RoboRio