PopupDetection Classes

The PopupDetection classes are also divided to view, model and viewModel.

When the user wants to see the detections of his flight, according to another regular flight, the PopupDetection classes creates a viewList with the detections.

The shown list contains the last 10 detections that were found at the flight so far.

As for the MVVM:

Every detection describes with: Feature1, Feature2 (the correlated features that raised the anomaly) and the AnomalyTime (the time of the detection).

When an anomaly is detected, the model informs the viewModel about the detection using the NotifyPropertyChanged method.

Between the ViewModel and the View there is Data Binding, so whenever there is a detection, the viewModel informs to the view.

In the view the detection is added to the view list of the detections that represents.

The anomalies are detected using an anomaly detector algorithm that is written outside of the application.

The model is the only one that is in contact with this algorithm ant delivers forward the anomalies information by notifying.

The anomaly detector algorithm is encapsuled in a dynamic-link-library (DLL) and is not known to the model in compilation time.

As long as the DLL implements a specific few functions, the model can use it even though he gets the DLL only in runtime .

Also, in the same way, the user can switch algorithms even when the flight is already running. We supplied to algorithm for the user to choose - one is detecting anomalies based on linear regression and the second detects using inner circle.

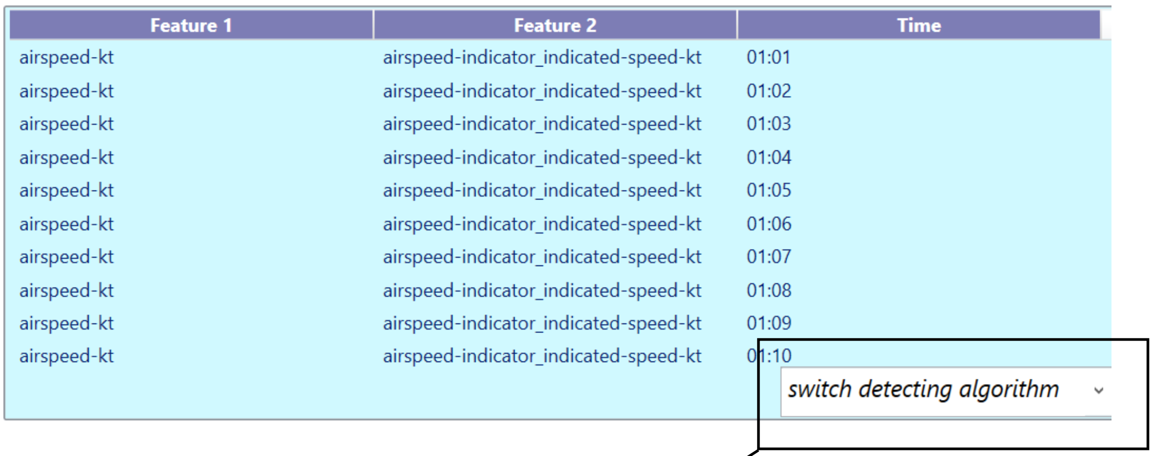
So that if the user wants to detect anomalies in the flight, he will not have to write himself an algorithm. But, if the user do want to use another algorithm he simply can do it in the "Add.." button and choose his algorithm from his folders with the file dialog.

This is possible before the flight starts and also during the flight later.

Note that if the user chooses in the menu that he wants to see the anomalies during the flight, the is an option that pops to him so he could choose the algorithm as described above and also upload a csv file with data of a normal flight so that the algoritm can learned it and later detect anomalies in the broadcasted flight that deviate from the normal data.

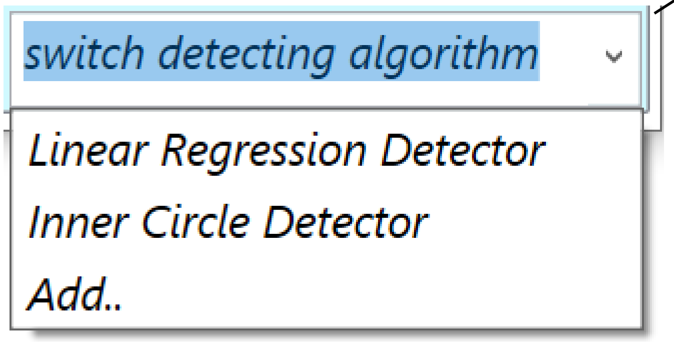
The data of the flight is streaming in a pulse of 10 csv lines in second, so in one second there can be 10 alarms of the same anomaly. We handled this issue by checking if the anomaly is in the same second and the same correlated featured with another one that is

already appears on the screen and if so, it would not show it again.



Feature 1	Feature 2	Time
airspeed-kt	airspeed-indicator_indicated-speed-kt	01:01
airspeed-kt	airspeed-indicator_indicated-speed-kt	01:02
airspeed-kt	airspeed-indicator_indicated-speed-kt	01:03
airspeed-kt	airspeed-indicator_indicated-speed-kt	01:04
airspeed-kt	airspeed-indicator_indicated-speed-kt	01:05
airspeed-kt	airspeed-indicator_indicated-speed-kt	01:06
airspeed-kt	airspeed-indicator_indicated-speed-kt	01:07
airspeed-kt	airspeed-indicator_indicated-speed-kt	01:08
airspeed-kt	airspeed-indicator_indicated-speed-kt	01:09
airspeed-kt	airspeed-indicator_indicated-speed-kt	01:10

switch detecting algorithm


switch detecting algorithm
Linear Regression Detector
Inner Circle Detector
Add..