https://www.garot.com/trading/

01 – Introduction

This is the informal video blog.

In part 4 of this VLOG series, we will build upon the Expert Advisor (EA) code from part 3.

I call this part: **Finishing the EA Code**

(page 2)

In this VLOG we will:

- Detect when pending order was filled
- Delete the remaining pending order
- Create a Fixed Trailing SL to let profits ride

Now let's take a look at the additions to the code.

02 – Code View in WinMerge

To show the additions to the code from version 03 to 04, I will do a quick walk-thru summary using WinMerge. After this summary, I will go through the added code in more detail in MetaEditor.

Remember that in WinMerge changed code is highlighted yellow on both sides, and additions (new code) is gray on the left side and yellow on the right side.

We first see the addition of the fixed trailing points (AR_FIXED_TRAILING_PTS).

Then we see that FixedTrailingStop() is no longer commented.

We introduce the OnTradeTransaction() event, which will tell us when one of our pending orders is filled.

FixedTrailingStop() keeps the SL a maximum defined distance from prices.

I collapsed the broken out code back to a single line using the ternary operator. For me, this compact version is simpler.

OrderDeletePending() deletes all pending orders.

PositionModify(), in our EA, changes the SL.

03 – Code View in MetaEditor

I'm back in MetaEditor, and we're looking at the code for Rapid Doji EA 04.

We first see the addition of the fixed trailing points variable (AR_FIXED_TRAILING_PTS). The AR_ at the beginning stands for "Adjust Risk," for I have additional methods to adjust risk, and I like to keep these variables together. The _PTS at the end stands for "points." In this case, I have a default value of 2700 points, which for many ForEx pairs equates to 270 pips. Essentially, this variable says that we will keep the SL no more than 2700 points away from prices. This value will be different for different symbols and different time frames. For the daily Pound-Dollar (GBPUSD), this can be anywhere from 1

to 3 ATR depending upon the volatility of the market. This points out something immediately: a fixed trailing SL does not consider the volatility of the market. Thus, we may, in future, desire create a trailing SL based upon ATR.

Note also that the fixed trailing points (AR_FIXED_TRAILING_PTS) variable is an input variable. That means we can vary this parameter using the Strategy Tester.

Now we jump to the OnTick() event where we see that the call to FixedTrailingStop() is no longer commented. Note that the function is called only on a new bar when a position is open. Thus, for this trailing stop method, we only adjust risk each bar. There are many other ways a trailing stop can be designed—this is just one possibility. Let's take a look at the code now.

The FixedTrailingStop() function first calculates the step amount, which is the distance or difference from the current prices to the maximum desired SL distance. For this trivial example of a fixed amount, simply multiply by the point size.

Here we define the price variable, which I initialize to a nonsense value. As I have mentioned before, this is a software development technique I use to quickly have errors bubble up and indicate where I may have made a mistake.

These constants are short-hand to make the code more readable.

In this code block, I determine whether the open position is currently LONG or SHORT. Based upon this, I can determine:

- the appropriate price to use
- our current risk, which is the SL difference to the current price
- our profit or loss, which is the distance between the current price and the position's open.

My first test checks if the distance from price to SL is greater than our step amount. If it is, I then check if the current profit is greater than the spread. In my thinking, there is no reason to move the SL if we aren't profitable.

This code block determines the new SL. Based upon whether we are LONG or SHORT determines if we add or subtract the step amount.

If we dropped down to this point, we request to modify the position with the new SL. Here I pass a NULL for the TP, which tells the code not to change the TP, which wasn't set anyway.

The function PositionModify() is a wrapper around the CTrade standard library PositionModify() method. I pass in the symbol, new SL, and NULL for the TP.

Now let's take a look at the issue we saw in the previous video—how to delete the alternate pending order from the one that was filled. We'll jump to the OnTradeTransaction() event.

The OnTradeTransaction() event runs when the TradeTransaction event occurs. This event occurs in a variety of cases, which you can look up in the documentation. For our purposes, we want to detect when a pending order activates on the server.

So, the server notifies MT5 when every TradeTransaction event occurs. There can be many of these, so it is our job to determine the specific event that denotes that an order has converted to a position.

The first thing we do is ensure the "deal" is non-zero. A "deal" is the result of an execution of an order. With a non-zero deal transaction, we can use HistoryDealSelect() to select this transaction. "Selecting" in this context means that we are pointing at this deal such that we can "work with it further," which simply means other related function we call will reference this specific deal.

Since an order can have multiple “deals” associated with it, our next step is to detect if this particular transaction is the one we want. The way to determine if a pending order has been filled is to detect if a DEAL_ENTRY of DEAL_ENTRY_IN occurred.

If that’s the case, let’s delete all other pending orders. Of course, in our case, there is only one other pending order.

Let’s look at OrderDeletePending().

This function is essentially a wrapper around the OrderDelete() method of the CTrade standard library.

This function was written for the generic case of wiping out all existing pending orders. It’s a bit like using a sledge hammer instead of a scalpel, but it’s simple and effective.

We loop over all existing orders backwards. We do this backwards because removing an item from the list changes the list dimension, which would make the looping index go out of whack.

We select the current order to work on it further, meaning that related functions will point to this specific order.

We then ensure that this order is for the correct symbol and “one of ours” by checking the magic number.

Finally, I check the order type against the list of known pending orders. Again, this function takes a sledge hammer approach, meaning that I wipe out ALL pending orders even though this particular EA only places BUY_STOP and SELL_STOP orders.

04 – Running In The Tester

Now I am back in MetaTrader 5. I will open the Strategy Tester by pressing CTRL+R.

I want to Visualize, and I will use similar parameters that I used in the previous video.

This time I want to check the [Inputs] tab to see the input parameters I have set. These all look fine to start.

After hitting the [Start] button, we got some actual trades, and the first two are profitable!

If I scroll to the bottom of the [History] tab, I see that we are are profitable over the entire testing period of five years.

Now let’s take a look at the first trade by double clicking on deal 2 order 2. Scroll back just a little to see the trade ENTRY.

We see:

1. the yellow arrow pointing to our Doji
2. the trigger prices at the high and low of the Doji bar
3. the initial SL positions at 1.75 * the ATR.

Note that the initial SL uses a different algorithm than the trailing SL.

The very next bar triggers a BUY order and we go LONG. If you remember from the last video, this same trade exited at the trigger price for the SELL_STOP, but this time, we don’t have that problem because we have deleted that pending order. Excellent!

Scanning ahead, we see the SL goes up as the bars (that is prices) tend upward. If we measure from the close to the SL, we should get roughly 2700 points. And . . . we do. Excellent.

Finally, we see the position closes when it hits the final trailing stop. A SL is not the most subtle way to leave a trade, but as you can see, it works.

Now let's take a look at our first failed trade by double clicking on the Deal 6 Order 9 trade. Scroll ahead a bit to see the Doji at which the trade was placed.

If we glance at what's happening, we see that prices are actually consolidating. Even while our trade is open we see multiple Dojis, which denote indecision. Eventually the market does go down again, but we get stopped out before enjoying a profit. Worse yet, we get another Doji, which triggers a downward trade a few bars before the local downward trend ends. So, we see two areas we might be able to improve:

1. Detect consolidation and tighten the stops.
2. Don't enter a trade on a local trend because it might have already spent it's momentum.

Of course, adding more "rules" increases the complexity of the system, which makes it less robust and leads to curve-fitting errors (issues).

05 – Wrapping up

(read from slide)

Final Notes (not in video)

1. A note on inconsistent variable names. AR_FIXED_TRAILING_PTS is all caps with underscores, which I typically reserve for constants. Yet this variable is an input variable. I would normally use mixed case and start with the letters Inp, e.g. InpFixedTrailingPts. Why the difference? This is for historical reasons. I have about a dozen Adjust Risk methods, and they all begin with AR_. I should have changed the name of this variable for these videos, but I just noticed it, and three videos are already done.
2. I use the capital D in a variable name to indicate a points difference. To convert to points count, I would divide by the point size. So, for example, I have AR_FIXED_TRAILING_PTS set to 2700 points. The current GBPUSD price is 1.29157. Obviously, I cannot add 2700 directly to 1.29157. So, I multiply by the point size, which is 0.00001, or Dpts = 0.02700.
3. If you have any suggestions for improvement to this script or this video series, do let me know. Programming is a bit of an art form, and each programmer will do it their own way.