The 3mTSHSUHC.*rp1*: Design and Optimization of Three-minute Timeframe Stoporder Hedging Strategy Using Heatmap Candles

Albeos Rembrant[12]*

[1]Founder & Chief Executive Officer
[2]Wildmind Quasars, Mangangalakal ng Kumikinang na Ginto
https://github.com/algorembrant/QAT-QuantitativeAlgorithmicTrading
December 3, 2025

Abstract

This study evaluates the intraday performance of the Three-minute Timeframe Stoporder Hedging Strategy Using Heatmap Candles (3mTSHSUHC). The model was tested using 500 three-minute candles, representing a single intraday dataset. The strategy relies on a heatmap candle framework that combines up/down tick-volume bars and high–low 0.25-tick-based range candles, with volume weighted on a 0–1 scale to identify high-volume-activity candles. A trade signal is generated when a candle's scaled volume exceeds 0.60, triggering a metric spike. Following each signal, the strategy places simultaneous buystop and sellstop orders at the candle body's high and low, respectively. Stop-loss is fixed at 5 price units (500 pips), while take-profit varies dynamically by the next metric spike event.

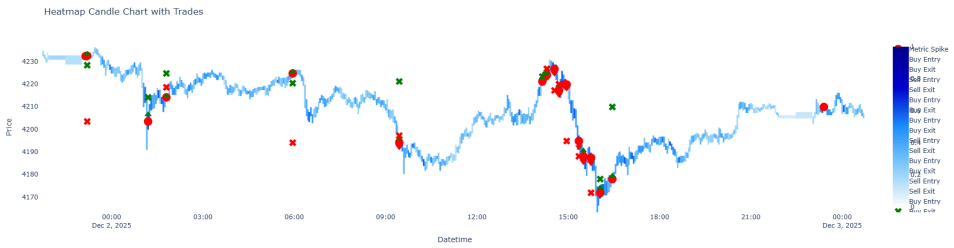

Figure 1: *Heatmap Candles Chart with Trade Executions*

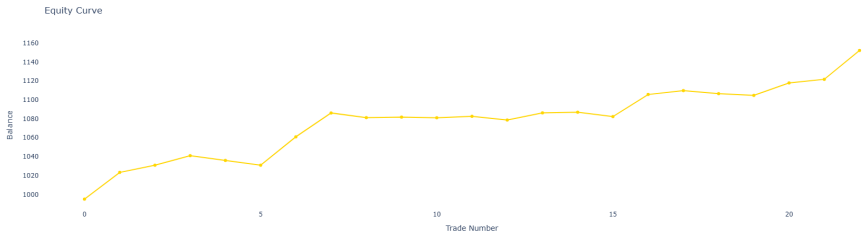

Figure 2: *Equity Curve of 3mTSHSUHC's Performance*

Results indicate strong short-term performance: with an initial balance of 1,000 units and 1% risk per trade, the model achieved a 15.22% return, 60% win rate, 1% maximum drawdown, and a Sharpe ratio of 8.48 across 23 trades under intraday period. However, the findings are constrained by the limited dataset and unoptimized metric-scaling mechanics, which may influence the strategy's consistency under different market conditions. Broader datasets and refined metric calibration are recommended for future research to assess the robustness and generalizability of the 3mTSHSUHC model.

Introduction

The study aims to answer the following questions:

1. How well did the Three-minute Timeframe Stoporder Hedging Strategy Using Heatmap Candles (3mTSHSUHC) performed during intraday. Terms of,
    a. Trades
    b. Winrate %
    c. Return %
    d. Max drawdown %
    e. Sharpie Ratio (price units)
    f. Highest R-multiple
    g. Lowest R-multiple
    h. Average Profit %
    i. Average Loss %
    j. Maximum Take Profit (price units)
    k. Maximum Stoploss (price units)

Methodology

Data Set

There will be only an intraday worth of data set. Five-hundred three-minute timeframe candles (500 3m candles) will be used for the test. This study is the 1st-version of the said strategy model, the metric scaling corresponding to trade execution has not been optimized yet and varies to the number of lookback as the scale changes dynamically. For further details, look at the Recommendation section of this study.

Heatmap Candles Chart

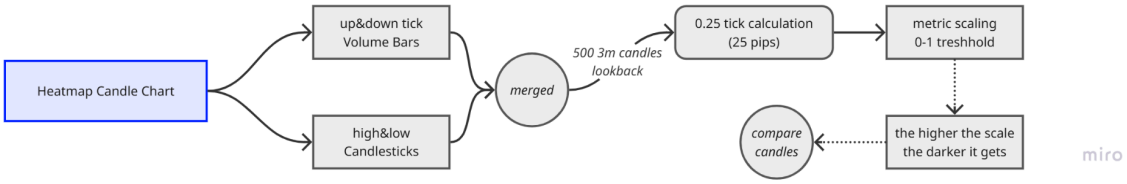

Figure 3: *Flow and process of Heatmap Candle calculation*

The heatmap candles chart is made from combining up & down tick volume based bars (the one on the very bottom of the chart) and high & low candle 0.25-tick-based layout (range candlesticks). The heatmap calculation is limited to 500 3-minute candles (accounting intraday profile) due to the metric scaling mechanic. The more bars it calculates, the more volume the metric will hold, and may result in lesser trade signals. That is why we will be only using an intraday dataset. The metric calculates the volume from the number of lookback, it weights and scales 0 to 1, zero as the lowest volume and one as highest volume. The higher the volume, the closer to scale of one, and the darker the color of the heatmap candles.

Figure 4: *Intraday data set with Heatmap Candles Chart*

Trade Signal and Execution

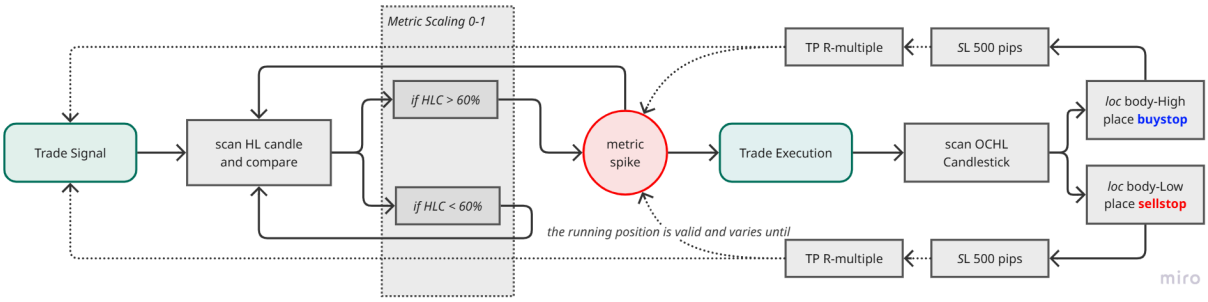

Figure 5: *Flow and process of Trade signals and Executions*

Trade Signal

There will be only one possible signal. It came from comparing the single candle's HL volume among the rest. If the volume (metric scaled to 0-1) is greater than 60% (volume > 0.6 scale), then a metric spike (red circle) will pop up at the center of the said candle. And if it's not greater than 60%, then continue scanning.

Trade Execution

Trade entry placements are executed after the metric spike. Scanning the candlestick's (OCHL) body, place a buystop at the body-high and sellstop order at body-low. The order placement does not include wick-based price. Moving on, the stop loss distance is 5 price units (500 pips) and the takeprofit distance will vary at where the next metric spike signal will pop up. Then the loop continues.

Results

Heatmap Candle Chart with Trade Executions

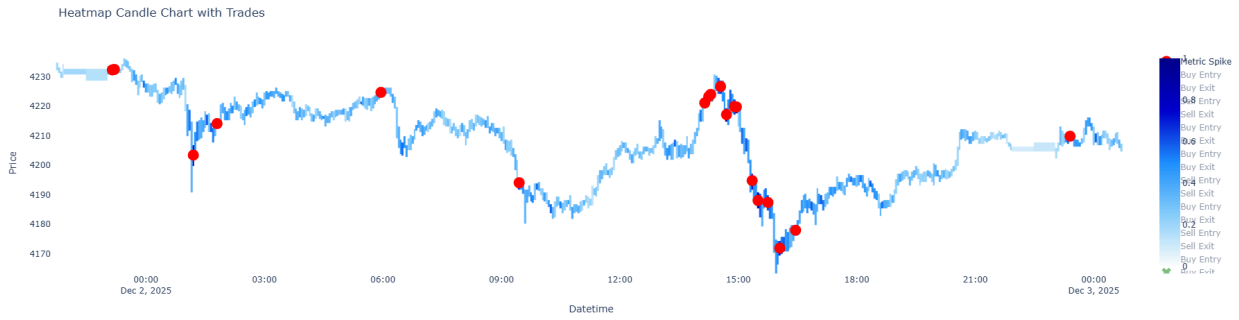

Figure 6: *Heatmap Candles Chart with Metric Spike signal*

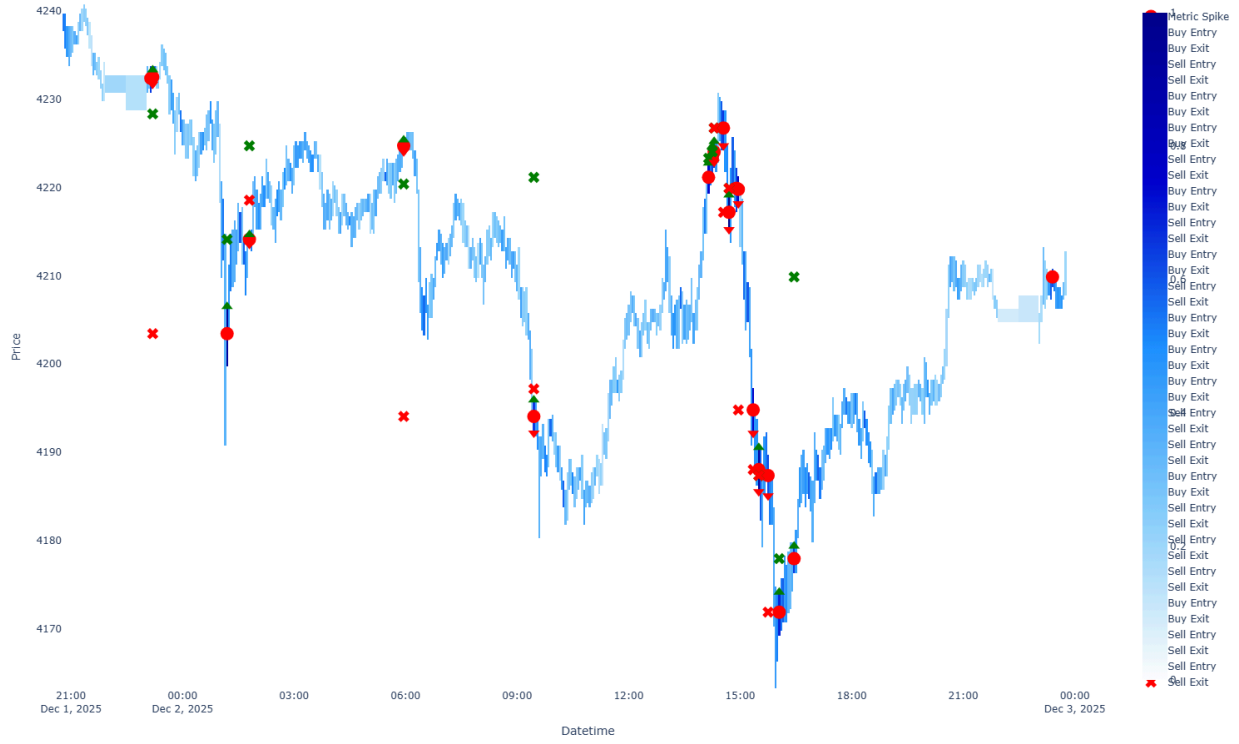

Figure 7: *Heatmap Candles Chart with Trade Executions*

Equity Curve

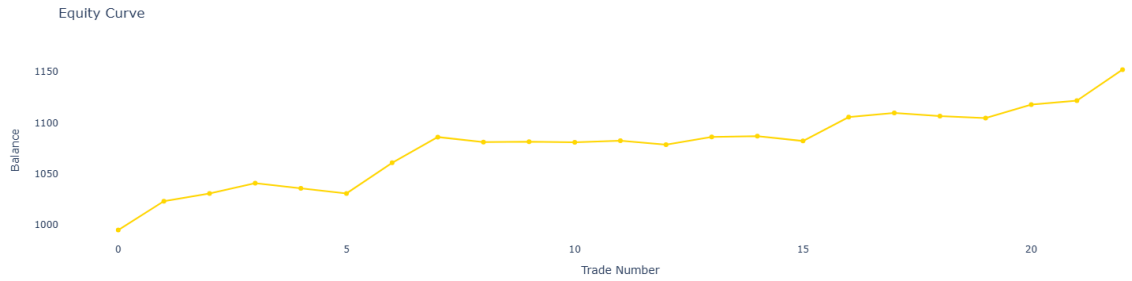

Figure 8: *Equity Curve of 3mTSHSUHC's Performance*

Stats

```
Initial Balance: 1000
Final Balance: 1152.23
Total Return %: 15.22%
Total Trades: 23
Win Rate: 60.87%
Max Drawdown: 10.00
Sharpe Ratio: 8.48
Highest R-multiple: 6.10
Lowest R-multiple: -1.00
Average Profit %: 1.33%
Average Loss %: -0.38%
Maximum Take Profit (price units): 30.50
Maximum Stoploss (price units): 5.00
```

Figure 9: *Statistical performance of 3mTSHSUHC*

Discussion

The test was done with an initial balance of 1000 (not specific in currency), and 1% bet as risk per trade from the latest capital. The results are considerably profitable, a total of 23 trades, return of 152.23 (15.22%), max drawdown of 10 (1%), win rate of 60%, sharpe ratio of 8.48, and other notable stats. The tests were done on intraday, meaning within a day from market open to market close of December 2, 2025 UTC+8.

Conclusions

The test shows promising results. With a 1,000 initial balance and 1% risk per trade, the strategy produced a 15.22% return, 60% win rate, low 1% maximum drawdown, and a Sharpe ratio of 8.48 across 23 trades. However, these outcomes are based on a single-day dataset, which limits the reliability and generalizability of the findings. A lack of broader historical data may change the model's performance when exposed to different market conditions.

Recomendations

Metric Scaling and Trade signal

The authors were unable to fully optimize the current metric calculation method. At present, the metric scaling relies on a 500-bar lookback, where a trade signal is triggered when candle volume > 0.60. However, adjusting the lookback period changes the effective volume threshold value, even with scaling applied. Future studies should therefore explore methods that prevent the metric from dynamically shifting with the lookback window, or develop alternative approaches for comparing a candle's volume relative to others more consistently.

Simulated Parameters

The authors did not incorporate spread and commission during the testing period. In the XAUUSD (Gold Forex) market using the Exness broker, typical conditions include a 0.3 spread (30 pips) on standard cent accounts or 0.2 spread (20 pips) on standard dollar accounts and a 1% commission on zero-spread accounts. These factors, along with realistic slippage and latency should be implemented in future simulations. Although the current test used a relatively wide stoploss of 5 (500 pips) and a small dataset, minimizing the immediate impact of such costs, future tests should account for them to ensure more accurate performance representation.

Backtesting

The trade execution component of the strategy functions as intended; however, issues arise in the generation of trading signals during backtesting. As previously noted, the metric scaling is based on all 500 three-minute candles simultaneously, meaning the calculation unintentionally incorporates future data relative to each metric spike. This creates a look-ahead bias, making the backtest results less reliable, since in live market conditions the strategy cannot access information from upcoming candles. Although the metric scale remains within the normalized 0–1 range, the underlying volume distribution continuously shifts as new candles enter or exit the 500-bar window. This can lead to inconsistent or misleading signals. For example, a single candle with exceptionally high volume may dominate the scale, raising the effective threshold and significantly reducing the number of valid signals—or, in some cases, preventing any signals from appearing at all. Aside from this structural limitation in the metric-scaling mechanic, the authors have not identified any other significant issues in the backtesting phase.

More Data sets

Future research should incorporate multi-day, multi-week, and multi-market datasets with varying volatility environments and market sessions. This will better evaluate the model's adaptability, stability, and long-term profitability. Additional stress-tests, parameter sensitivity analyses, and robustness checks are recommended to reinforce the strategy's reliability across diverse market conditions.

Appendices

Code

```
import MetaTrader5 as mt5
import pandas as pd
import numpy as np
import plotly.graph_objects as go

# ------------------------------
# 1. Initialize MT5
# ------------------------------
if not mt5.initialize():
    print("MT5 initialization failed")
    quit()

symbol = "XAUUSDc"
timeframe = mt5.TIMEFRAME_M3
num_bars = 500

# ------------------------------
# 2. Fetch historical data
# ------------------------------
rates = mt5.copy_rates_from_pos(symbol, timeframe, 0,
num_bars)
df = pd.DataFrame(rates)
df["datetime"] = pd.to_datetime(df["time"], unit="s")

# ------------------------------
# 3. Build 0.5 footprint clusters
# ------------------------------
cluster_size = 0.5
df["low_cluster"] = np.floor(df["low"] / cluster_size) *
cluster_size
df["high_cluster"] = np.floor(df["high"] / cluster_size) *
cluster_size

cluster_rows = []
for i in range(len(df)):
    low = df["low_cluster"][i]
    high = df["high_cluster"][i]
    vol = df["tick_volume"][i]
    dt = df["datetime"][i]
    price_levels = np.arange(low, high + cluster_size,
cluster_size)
    for p in price_levels:
        cluster_rows.append([dt, p, vol / len(price_levels)])

cluster_df = pd.DataFrame(cluster_rows,
columns=["datetime", "price", "cluster_vol"])

# ------------------------------
# 4. Metric scale
# ------------------------------
max_vol = cluster_df["cluster_vol"].max()
cluster_df["metric"] = cluster_df["cluster_vol"] / max_vol

colorscale = [
    [0.0, "white"],
    [0.25, "#87CEFA"],
    [0.50, "#1E90FF"],
    [0.75, "#0000CD"],
    [1.0, "#00008B"],
]

# ------------------------------
# 5. Metric spike detection
# ------------------------------
metric_threshold = 0.60
df["metric_spike"] = df["datetime"].apply(
    lambda dt: cluster_df[cluster_df["datetime"] ==
dt]["metric"].max() >= metric_threshold
)
df["center"] = (df["high"] + df["low"]) / 2

# ------------------------------
# 6. Backtesting trades
# ------------------------------
initial_balance = 1000
balance = initial_balance
equity_curve = []
trades = []

spike_indices = df.index[df["metric_spike"]].tolist()

for i in range(len(spike_indices)-1):
    entry_idx = spike_indices[i]
    exit_idx = spike_indices[i+1]
    takeprofit_price = df["center"].iloc[exit_idx]

    # BuyStop logic
    entry_price = df["high"].iloc[entry_idx]
    stoploss = entry_price - 5
    executed = False
    for j in range(entry_idx + 1, exit_idx + 1):
        if df["high"].iloc[j] >= entry_price:
            executed = True
            break
    if executed:
        exit_price = takeprofit_price if takeprofit_price >
stoploss else stoploss
        pl = exit_price - entry_price
        balance += pl
        equity_curve.append(balance)
        trades.append({"type":"Buy", "entry_idx":entry_idx,
"entry":entry_price, "exit":exit_price,
                       "PL":pl, "R": pl/5, "TP_units":
exit_price-entry_price, "SL_units": entry_price-stoploss,
                       "PL%": pl/initial_balance*100})

    # SellStop logic
    entry_price = df["low"].iloc[entry_idx]
    stoploss = entry_price + 5
    executed = False
    for j in range(entry_idx + 1, exit_idx + 1):
        if df["low"].iloc[j] <= entry_price:
            executed = True
            break
    if executed:
        exit_price = takeprofit_price if takeprofit_price <
stoploss else stoploss
        pl = entry_price - exit_price
        balance += pl
        equity_curve.append(balance)
        trades.append({"type":"Sell", "entry_idx":entry_idx,
"entry":entry_price, "exit":exit_price,
                       "PL":pl, "R": pl/5, "TP_units":
entry_price-exit_price, "SL_units": stoploss-entry_price,
                       "PL%": pl/initial_balance*100})

trades_df = pd.DataFrame(trades)
if trades_df.empty:
```

```
    print("No trades executed")
    mt5.shutdown()
    exit()

# ---------------------------------
# 7. Compute metrics
# ---------------------------------
num_trades = len(trades_df)
win_trades = trades_df[trades_df["PL"] > 0]
loss_trades = trades_df[trades_df["PL"] <= 0]

win_rate = len(win_trades)/num_trades*100
max_dd = (pd.Series(equity_curve).cummax() -
pd.Series(equity_curve)).max()
sharpe_ratio =
(trades_df["PL"].mean()/trades_df["PL"].std()*np.sqrt(252)) if
len(trades_df)>1 else 0
highest_R = trades_df["R"].max()
lowest_R = trades_df["R"].min()
avg_profit_pct = win_trades["PL%"].mean()
avg_loss_pct = loss_trades["PL%"].mean()
max_tp_units = trades_df["TP_units"].max()
max_sl_units = trades_df["SL_units"].max()
total_return_pct = (balance -
initial_balance)/initial_balance*100

# ---------------------------------
# 8. Print metrics
# ---------------------------------
print(f"Initial Balance: {initial_balance}")
print(f"Final Balance: {balance:.2f}")
print(f"Total Return %: {total_return_pct:.2f}%")
print(f"Total Trades: {num_trades}")
print(f"Win Rate: {win_rate:.2f}%")
print(f"Max Drawdown: {max_dd:.2f}")
print(f"Sharpe Ratio: {sharpe_ratio:.2f}")
print(f"Highest R-multiple: {highest_R:.2f}")
print(f"Lowest R-multiple: {lowest_R:.2f}")
print(f"Average Profit %: {avg_profit_pct:.2f}%")
print(f"Average Loss %: {avg_loss_pct:.2f}%")
print(f"Maximum Take Profit (price units): {max_tp_units:.2f}")
print(f"Maximum Stoploss (price units): {max_sl_units:.2f}")

# ---------------------------------
# 9. Plot footprint + trade markers
# ---------------------------------
fig = go.Figure()

# Heatmap footprint
fig.add_trace(
    go.Heatmap(
        x=cluster_df["datetime"],
        y=cluster_df["price"],
        z=cluster_df["metric"],
        colorscale=colorscale,
        zmin=0,
        zmax=1,
        showscale=True,
        name="Footprint Metric"
    )
)

# Metric spike markers
spike_rows = df[df["metric_spike"]]
fig.add_trace(
    go.Scatter(
        x=spike_rows["datetime"],
        y=spike_rows["center"],
        mode="markers",
        marker=dict(size=16, color="red", symbol="circle"),
        name="Metric Spike"
    )
)

# Trade markers
for _, trade in trades_df.iterrows():
    color = "green" if trade["type"]=="Buy" else "red"
    symbol_entry = "triangle-up" if trade["type"]=="Buy" else
"triangle-down"
    symbol_exit = "x"
    dt_entry = df["datetime"].iloc[trade["entry_idx"]]
    # Entry
    fig.add_trace(
        go.Scatter(
            x=[dt_entry],
            y=[trade["entry"]],
            mode="markers",
            marker=dict(size=12, color=color,
symbol=symbol_entry),
            name=f"{trade['type']} Entry"
        )
    )
    # Exit
    fig.add_trace(
        go.Scatter(
            x=[dt_entry],
            y=[trade["exit"]],
            mode="markers",
            marker=dict(size=12, color=color,
symbol=symbol_exit),
            name=f"{trade['type']} Exit"
        )
    )

fig.update_layout(
    title="Heatmap Candle Chart with Trades",
    xaxis_title="Datetime",
    yaxis_title="Price",
    plot_bgcolor="white",
    paper_bgcolor="white",
    width=1800,      # Increased width
    height=500,     # Increased height
)


# ---------------------------------
# 10. Plot equity curve
# ---------------------------------
fig_eq = go.Figure()
fig_eq.add_trace(go.Scatter(
    x=np.arange(len(equity_curve)),
    y=equity_curve,
    mode="lines+markers",
    line=dict(color="gold", width=2),
    name="Equity Curve"
))
fig_eq.update_layout(
    title="Equity Curve",
    xaxis_title="Trade Number",
    yaxis_title="Balance",
    plot_bgcolor="white",
    paper_bgcolor="white",
    width=1800,      # Increased width
    height=500,     # Increased height
)

fig.show()
fig_eq.show()
mt5.shutdown()
```