Automated Trading Bot: Feinkonzept

Juri Stoffers[a,1]

[a] *Gymthun*

Begleitperson: Dr. Ostrin Geoffrey

1. Define My Goals & Requirements

I start by deciding what kind of trading I want to pursue—momentum, arbitrage, or another strategy. I set clear targets for profits and acceptable losses, and choose the markets I'll focus on (crypto, stocks, or forex). I also consider performance requirements like speed, reliability, and implementation complexity.

2. Outline My System Design

2.1. Data Sources

I gather both real-time and historical market data to feed my bot, ensuring I have enough coverage for backtesting and live operations.

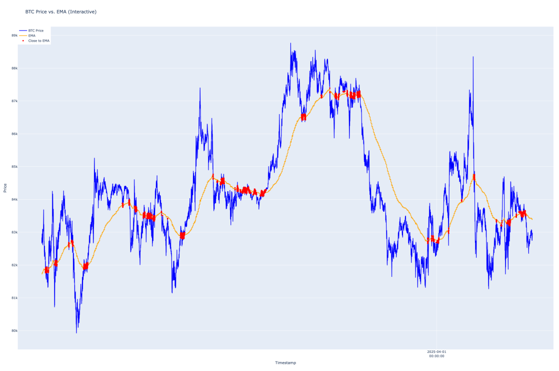

Figure 1. Current Data set visualization

2.2. Signal Processing

I develop modules to analyze data for trade signals, whether it's technical indicators (e.g., EMA, MACD) or more advanced methods (machine learning).

2.3. Efficient Market Hypothesis

Is it possible to abstract money from the market, because if I would find an edge, wouldn't it already be exploited so priced in?

2.4. Order Management

I connect to broker APIs so my bot can execute trades automatically. This includes handling order placement, cancellation, and partial fills.

2.5. Risk Management

Safety features like stop-loss orders and position sizing help me control potential drawdowns. I might use trailing stops or percentage-based position sizing.

2.6. Monitoring

I set up real-time tracking and logging to catch issues before they escalate, and to log trades for auditing performance over time.

3. Backtesting & Simulation

Before going live, I test my strategy on historical data to refine it and uncover potential flaws. I also run a simulated (paper-trading) environment to confirm the bot's logic without risking real funds.

3.1. Performance Thresholds

For example, a Sharpe ratio below 1.5 might be more realistic, whereas anything above that could indicate overfitting or unrealistic assumptions.

4. Implementation & Operation

I choose a programming language (e.g., Python) for its extensive libraries and community support. I build the system in modular parts so I can update individual sections without overhauling the entire setup. Finally, I integrate security measures like API encryption and strict access controls, while staying aware of regulatory requirements. This plan covers key areas: strategy, system design, testing, and secure live operation, giving me a roadmap for a trading bot that could potentially be profitable in real markets.

5. Feinkonzept Notes (Additional Topics)

5.1. Data Creation

- Data sources
- Creating a good timeseries dataset
- Hosting data creation process
- External datasources for out-of-sample testing

5.2. What is Bitcoin

A form of digital currency that uses blockchain technology to support transactions between users on a decentralized network.

5.3. How is the Price of Bitcoin Determined (Auction Principle)

Below is an example "orderbook" image illustrating how bids and asks are arranged in the marketplace:

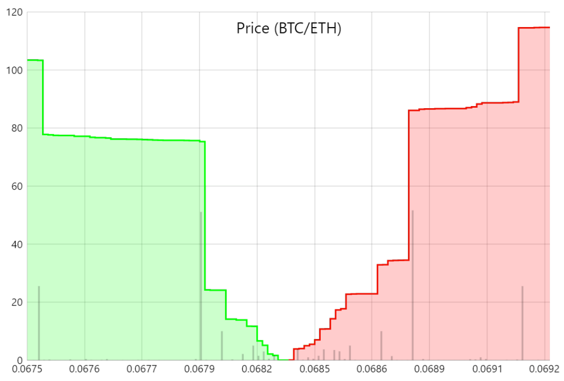

Figure 2. Example orderbook visualization, showing bid/ask levels.

5.4. Backtesting Details

- Testing different kinds of strategies (Sample vs Out-of-sample)
- Sharpe ratio threshold (e.g., below 1.5 = more realistic)
- Real-time paper trading test
- Outline which indicators I'll be using

5.5. Outlining My System Design (Extra Points)

- Programming language of choice (why?)
- Setting up system in modular parts
- Safety measures
- Reliability of external APIs

- Broker I'm planning to use
- Latency considerations
- Order management details
- Define running period for Matura levy, and timeline up to final presentation

6. Conclusion

This Feinkonzept provides a high-level plan for developing and deploying an automated trading system. Each section—goals, design, backtesting, and implementation—helps ensure both theoretical soundness and practical feasibility.