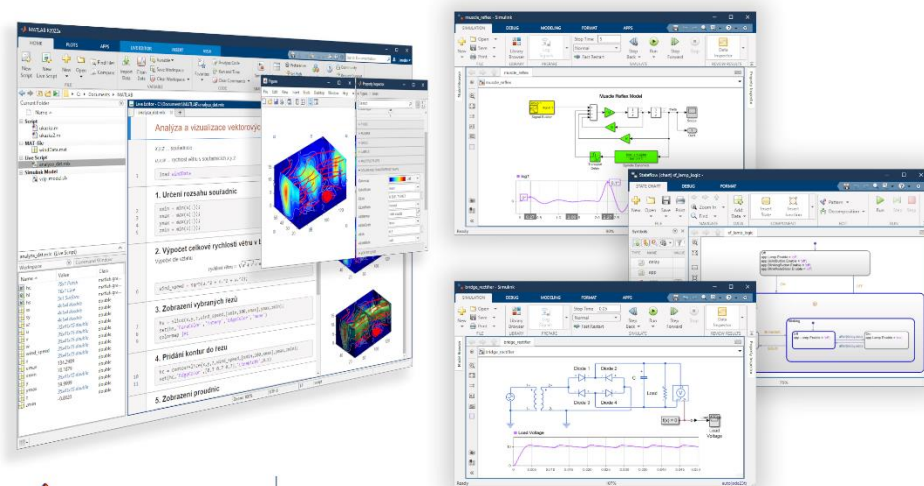


## AI and Model-Based Design

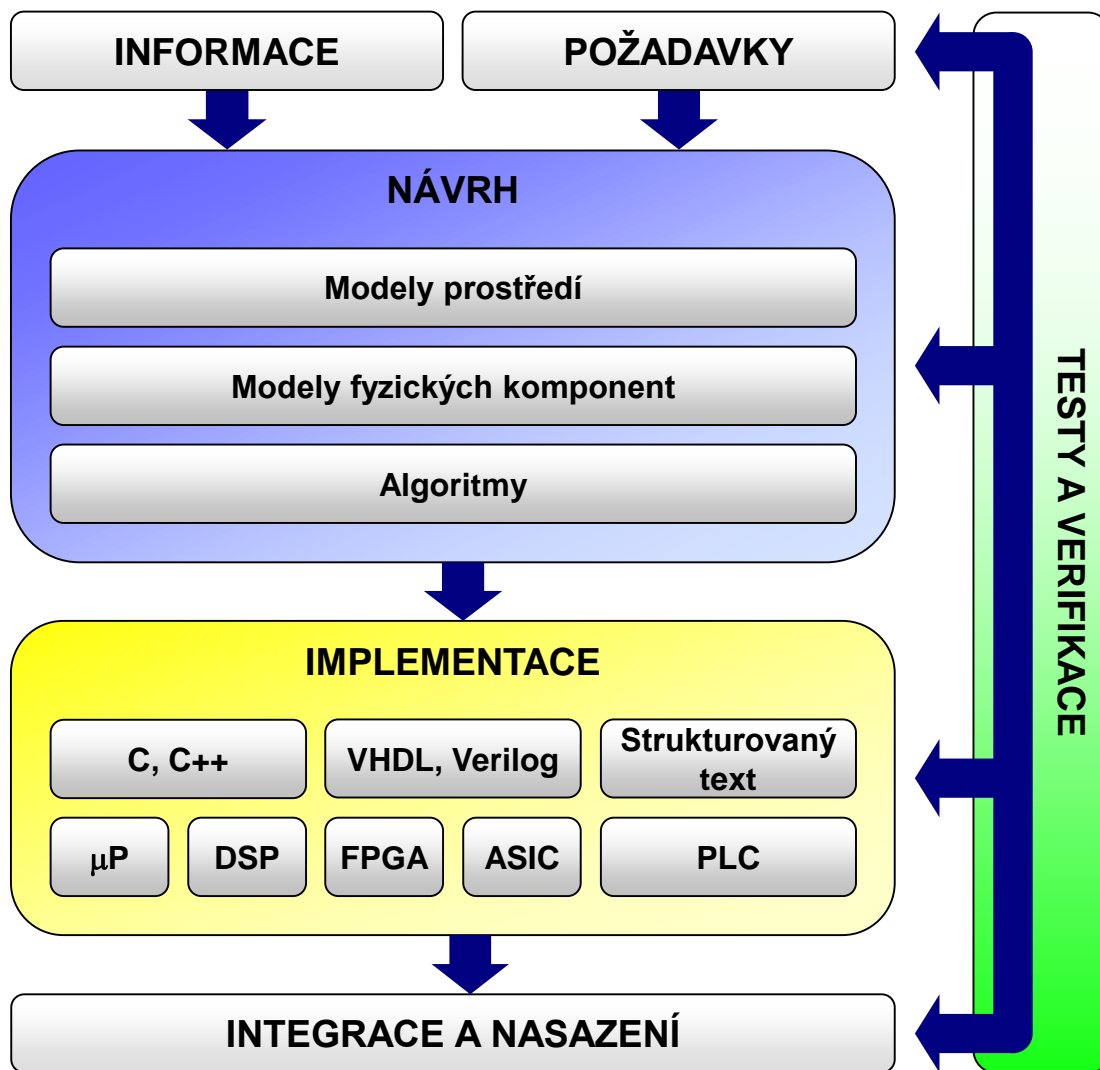


Jaroslav Jirkovský  
jirkovsky@humusoft.cz

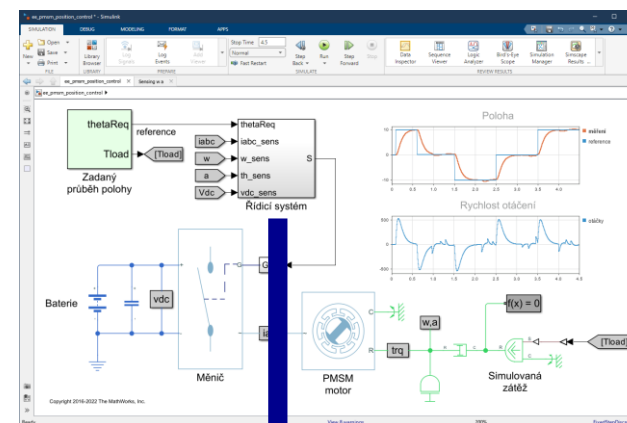
[www.humusoft.cz](http://www.humusoft.cz)  
[info@humusoft.cz](mailto:info@humusoft.cz)

[www.mathworks.com](http://www.mathworks.com)

# Vývoj metodou Model-Based Design



## Modelování, simulace a testování



Automatické generování kódu

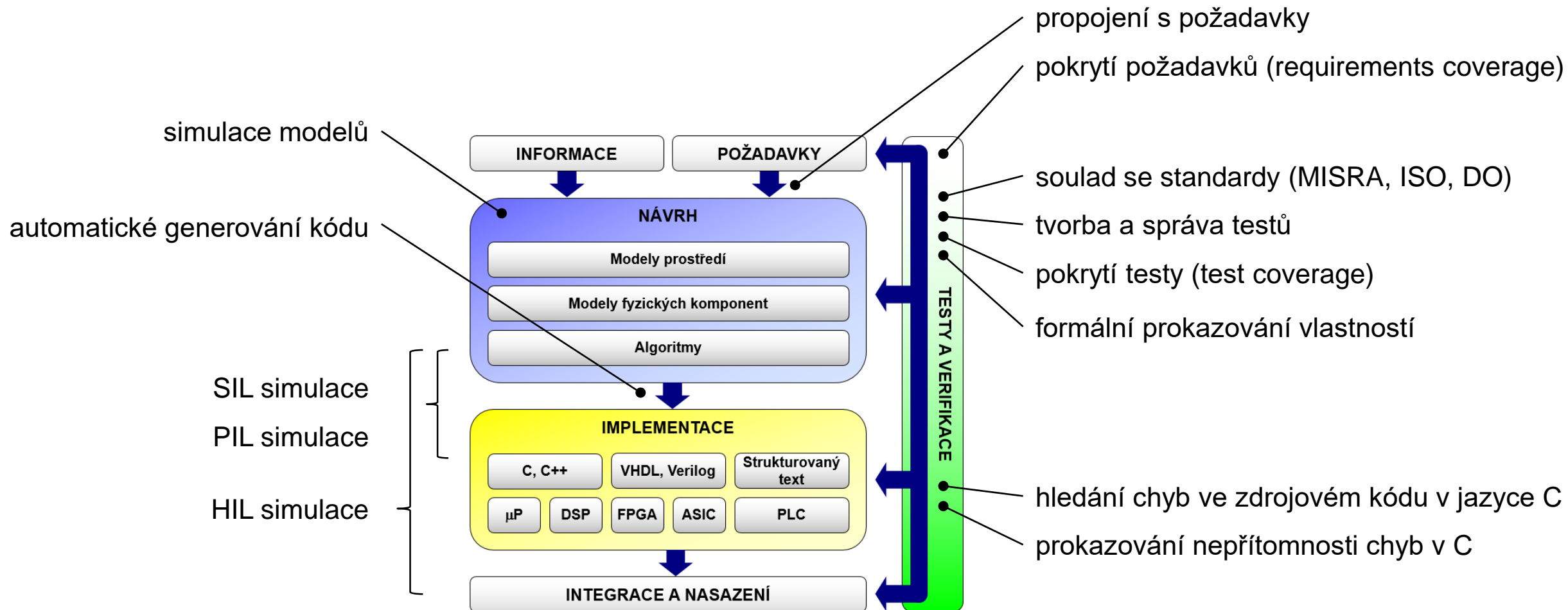
```

Code
Control_System.c
6 * Model version : 6.4
7 * Simulink Coder version : 9.9 (R2023a) 19-Nov-2022
8 * C/C++ source code generated on : Tue Apr 25 11:26:04 2023
9
10 * Target selection: ert-tlc
11 * Embedded hardware selection: Texas Instruments->C2000
12 * Code generation objectives: unspecified
13 * Validation result: not run
14
15
16 #include "Control_System.h"
17 #include <math.h>
18 #include "rt_udefints.h"
19 #include "rtwtypes.h"
20 #include <string.h>
21
22 /* Block signals (default storage) */
23 #blockio_control_system Control_System_B;
24
25 /* Block states (default storage) */
26 #blockstates_control_system Control_System_S;
27
28 /* External inputs (root import signals with default storage) */
29 ExternalInputs_Control_System Control_System_U;
30
31 /* External outputs (root outputs fed by signals with default storage) */
32 ExternalOutputs_Control_System Control_System_Y;
33
34 /* Real-time model */
35 static #if_HOML_control_system Control_System_H;
36 #if_HOML_control_system *const control_system_H = &Control_System_H;
37
38 /* Model step function */
39 void control_system_step(void)
40 {
41     real32_T B1AS;
42     real32_T uB;
43
44     _ATAPL282104_H801Podklady/control_system_ert_rtw/control_system.c Ln 6 Col 13
    
```

Nasazení




# Vývoj metodou Model-Based Design



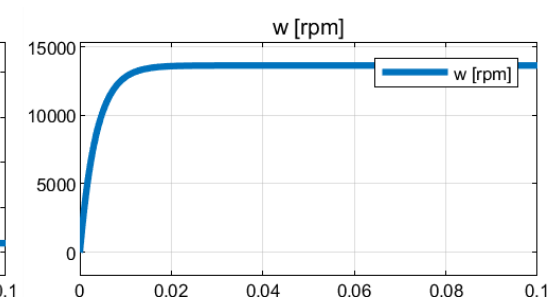
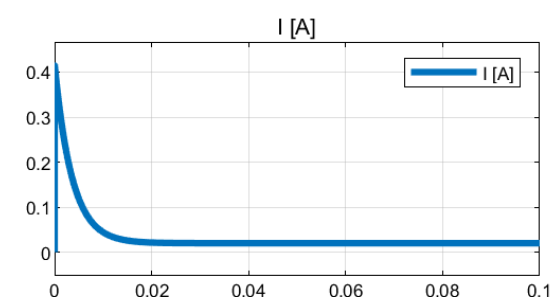
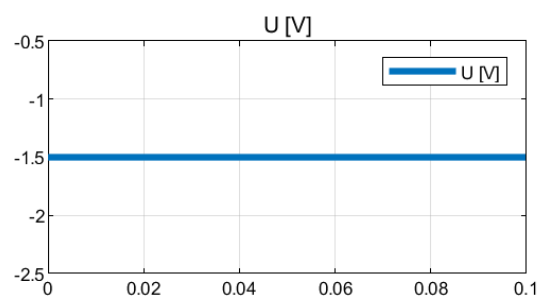
# System

## SYSTEM

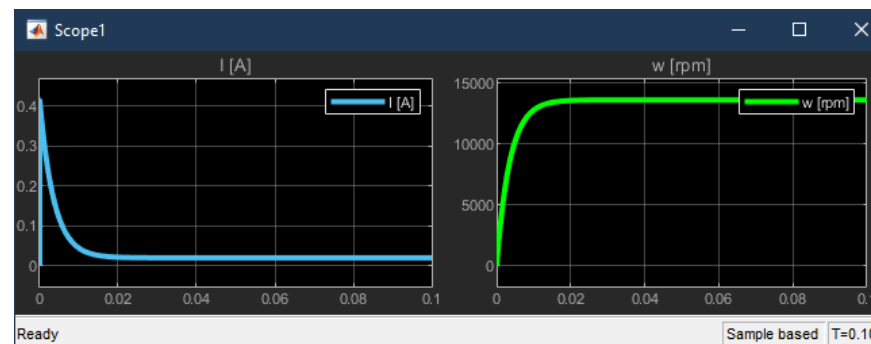
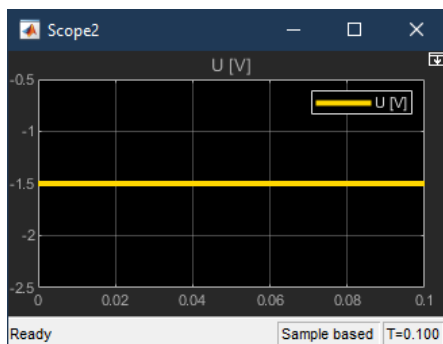
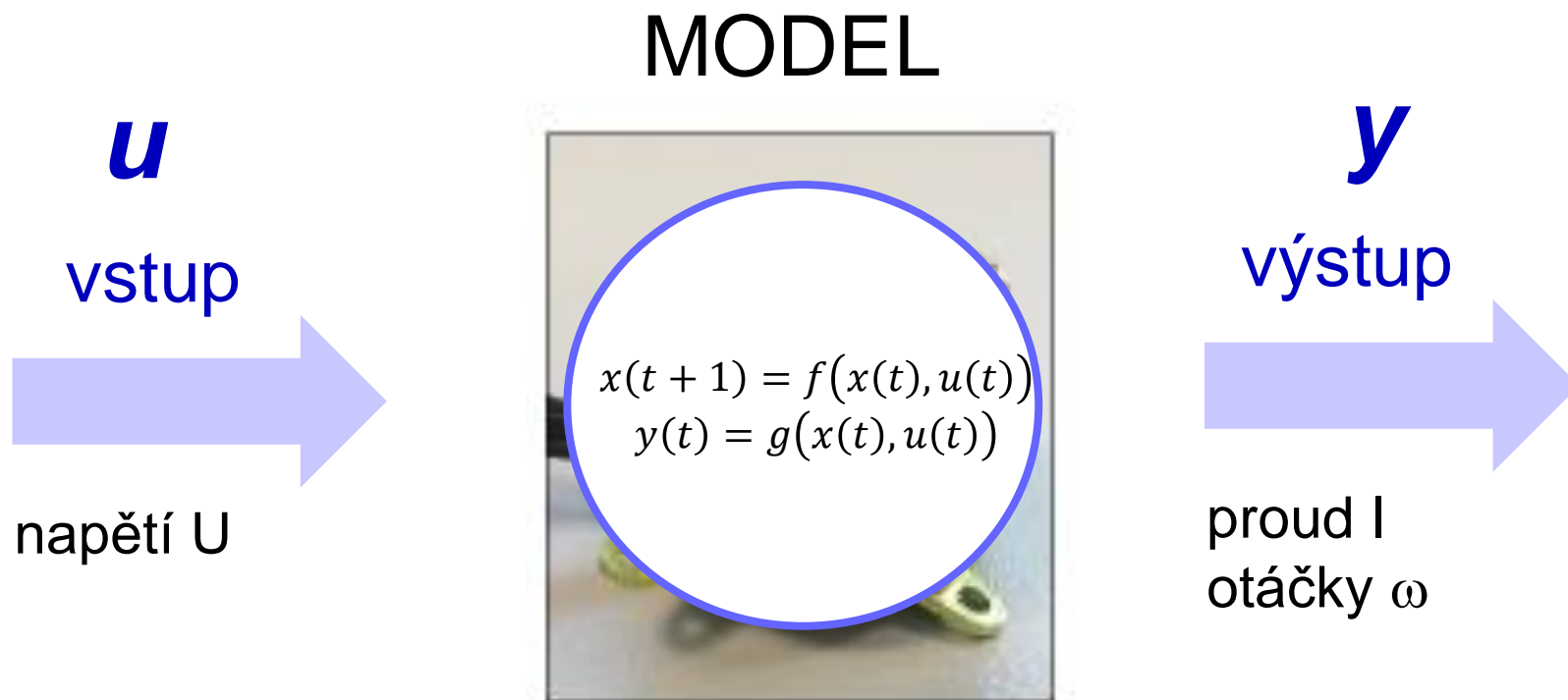
vstup  
  
 napětí  $U$



výstup  
  
 proud  $I$   
 otáčky  $\omega$



# Matematický model systému

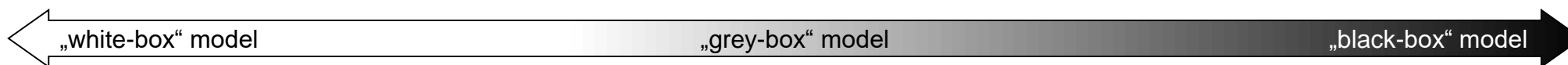


# Přístupy k modelování

- Pro různé situace jsou vhodné různé přístupy

## Fyzikální vztahy

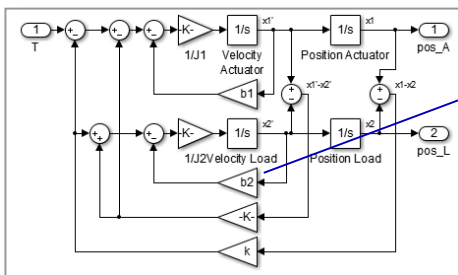
## Naměřená data



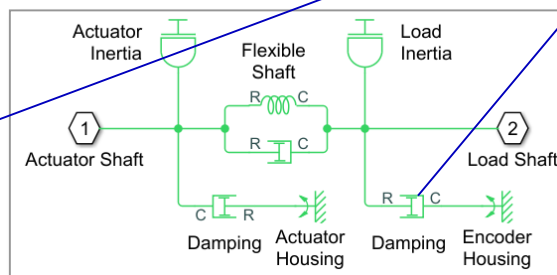
### Modelování rovnic

$$J_1 x_1'' = -b_1 x_1' - k(x_1 - x_2) - b_{12}(x_1' - x_2') + T$$

$$J_2 x_2'' = -b_2 x_2' + k(x_1 - x_2) - b_{12}(x_1' - x_2')$$

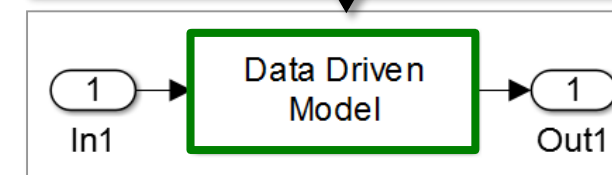
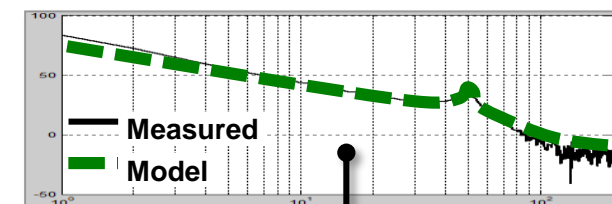


### Fyzikální síť



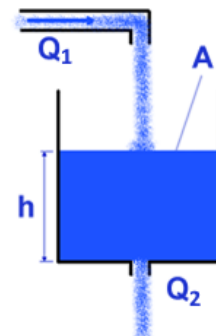
### Ladění neznámých parametrů

### Identifikace soustav

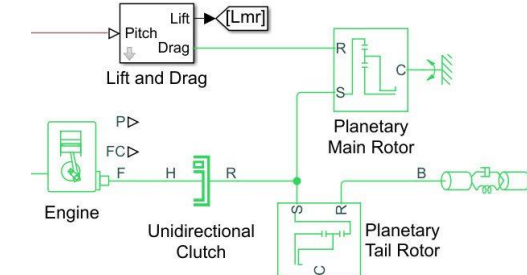
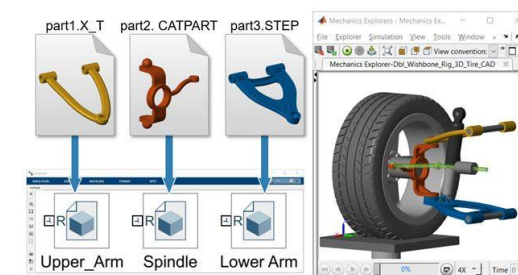
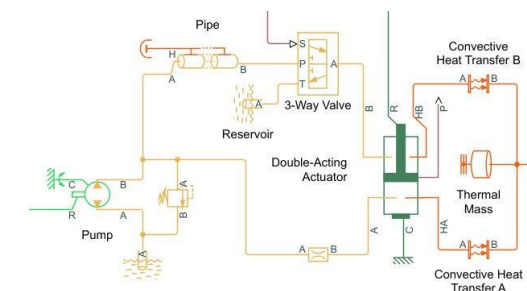
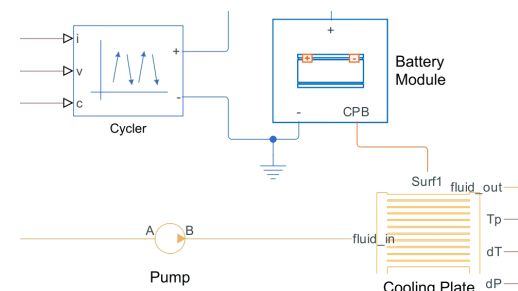
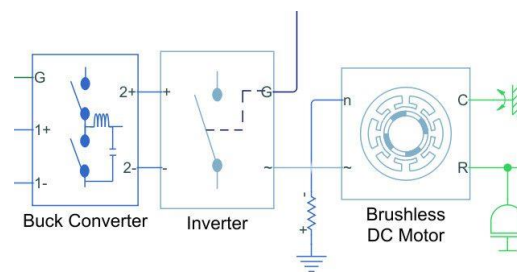
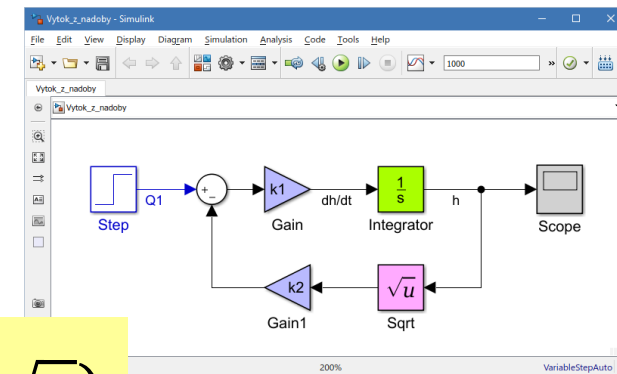


# Modelování soustav

- Popis matematickými rovnicemi
  - matematicko-fyzikální analýza
  - identifikace soustavy z naměřených dat
- Fyzikální modelování
  - elektromechanika
  - baterie
  - tekutinové systémy
  - 3-D mechanika
  - převodové systémy
- Aplicačně zaměřené
  - automobily, letadla, drony, roboty

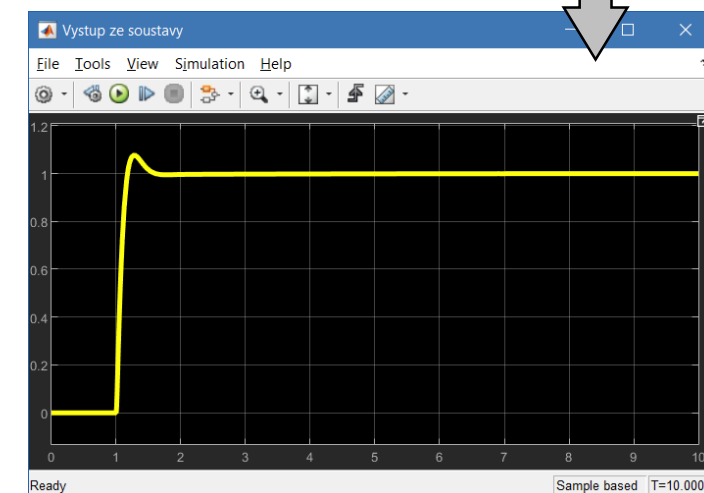
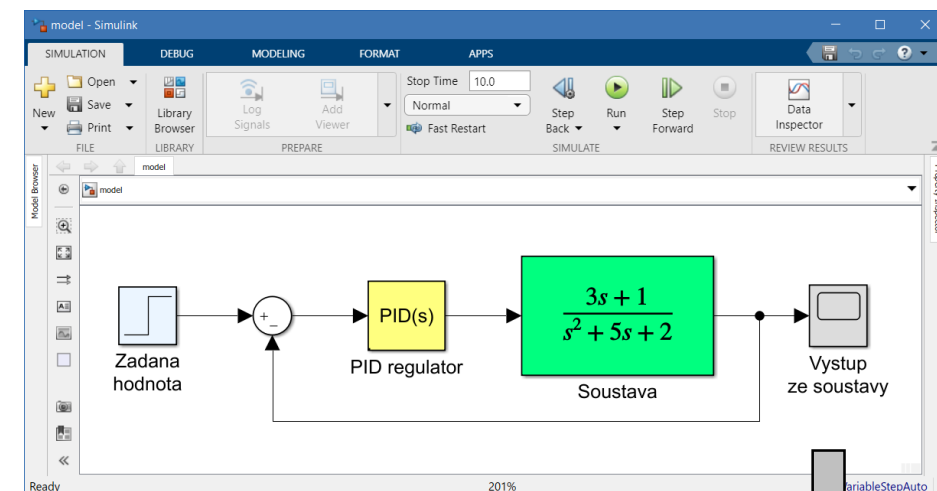


$$\frac{dh}{dt} = k_1(Q_1 - k_2\sqrt{h})$$



# Modelování algoritmů

- Řídicí systémy
- Zpracování signálu a komunikace
- Zpracování obrazu a počítačové vidění
- a další ...
  
- Společná simulace soustav a algoritmů
  
- Generování kódu pro cílové platformy
  - C/C++, HDL, PLC, CUDA





# AI

## UMĚLÁ INTELIGENCE (AI)

Libovolná technika, která umožňuje strojům napodobit lidskou inteligenci



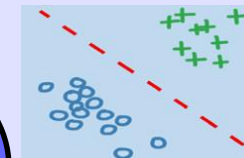
## MACHINE LEARNING

Statistická metoda, která umožní stroji "naučit se" zadanou úlohu na základě dat bez explicitní tvorby programu

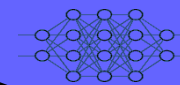
UNSUPERVISED LEARNING  
(neoznačená data)



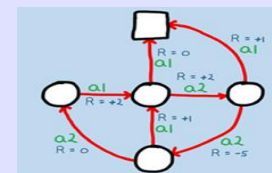
SUPERVISED LEARNING  
(označená data)



DEEP LEARNING  
(Neuronové sítě s mnoha vrstvami)



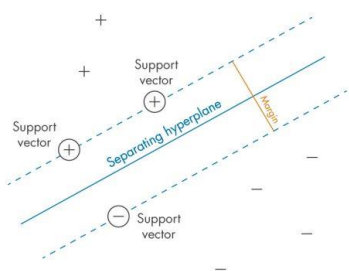
REINFORCEMENT LEARNING  
(data z interakce)



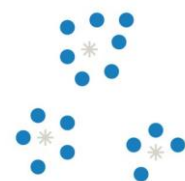
# AI modely v prostředí MATLAB

Machine learning

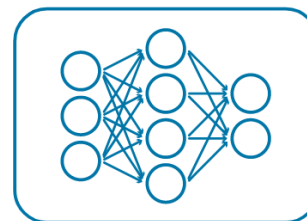
Deep learning



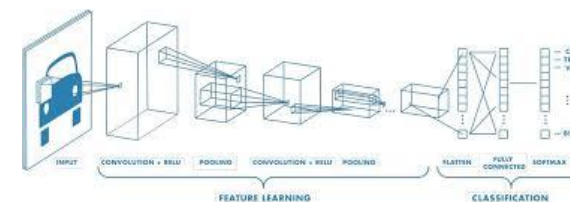
SVM



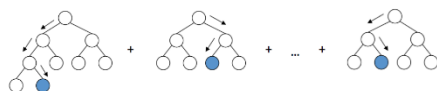
Clustering



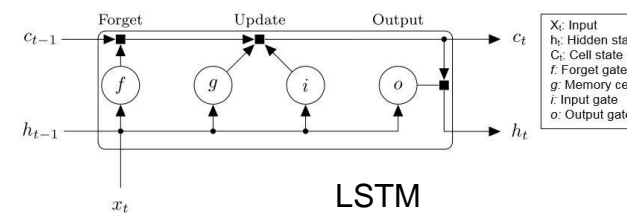
FC



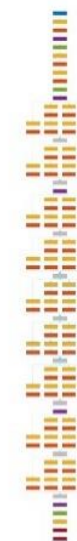
CNN



Decision trees



LSTM



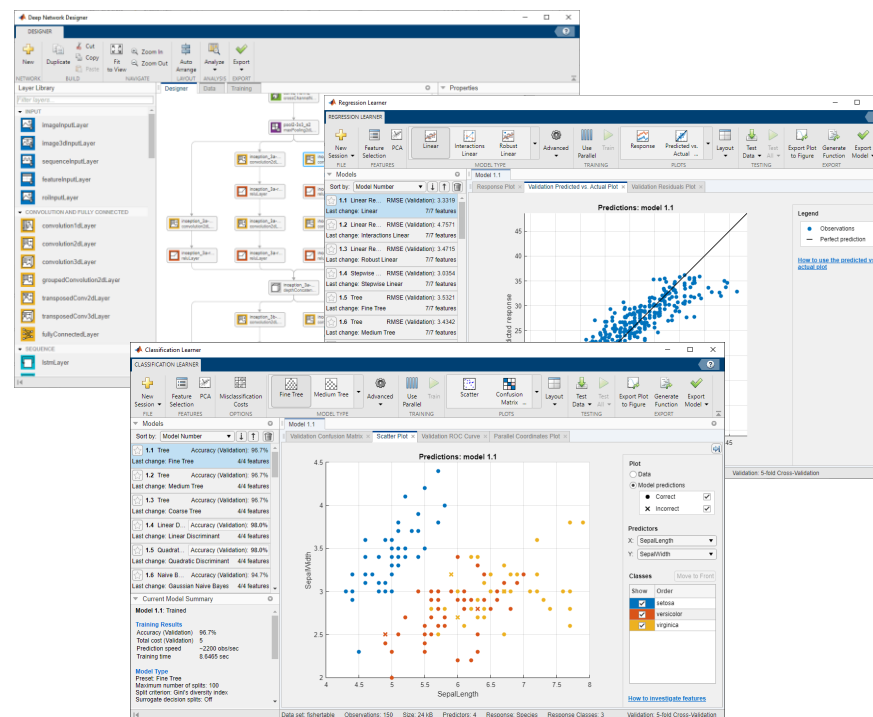
# 3 cesty k vytvoření AI modelu v prostředí MATLAB

```
inputSize = 12;
numHiddenUnits = 100;
numClasses = 9;

layers = [ ...
    sequenceInputLayer(inputSize)
    lstmLayer(numHiddenUnits, 'OutputMode', 'last')
    fullyConnectedLayer(numClasses)
    softmaxLayer
    classificationLayer]
```

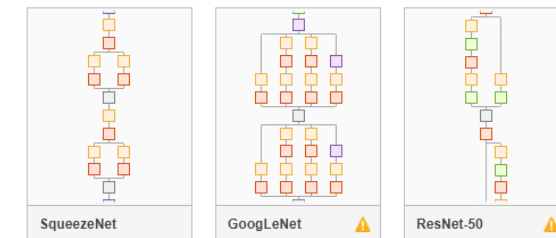
fitcauto / fitrauto

Funkce, skripty



Interaktivní návrh prostřednictvím grafických aplikací

## Image Networks (Pretrained)



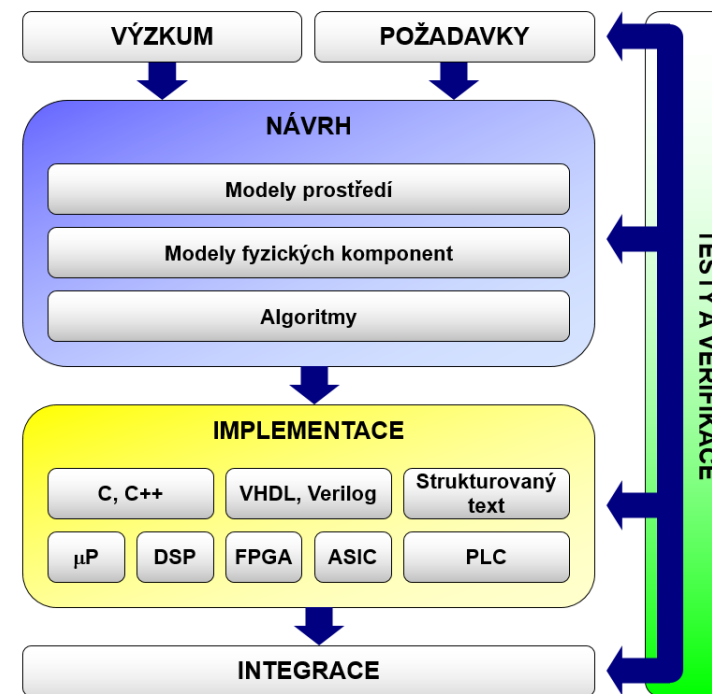
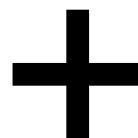
Show more

## Sequence Networks

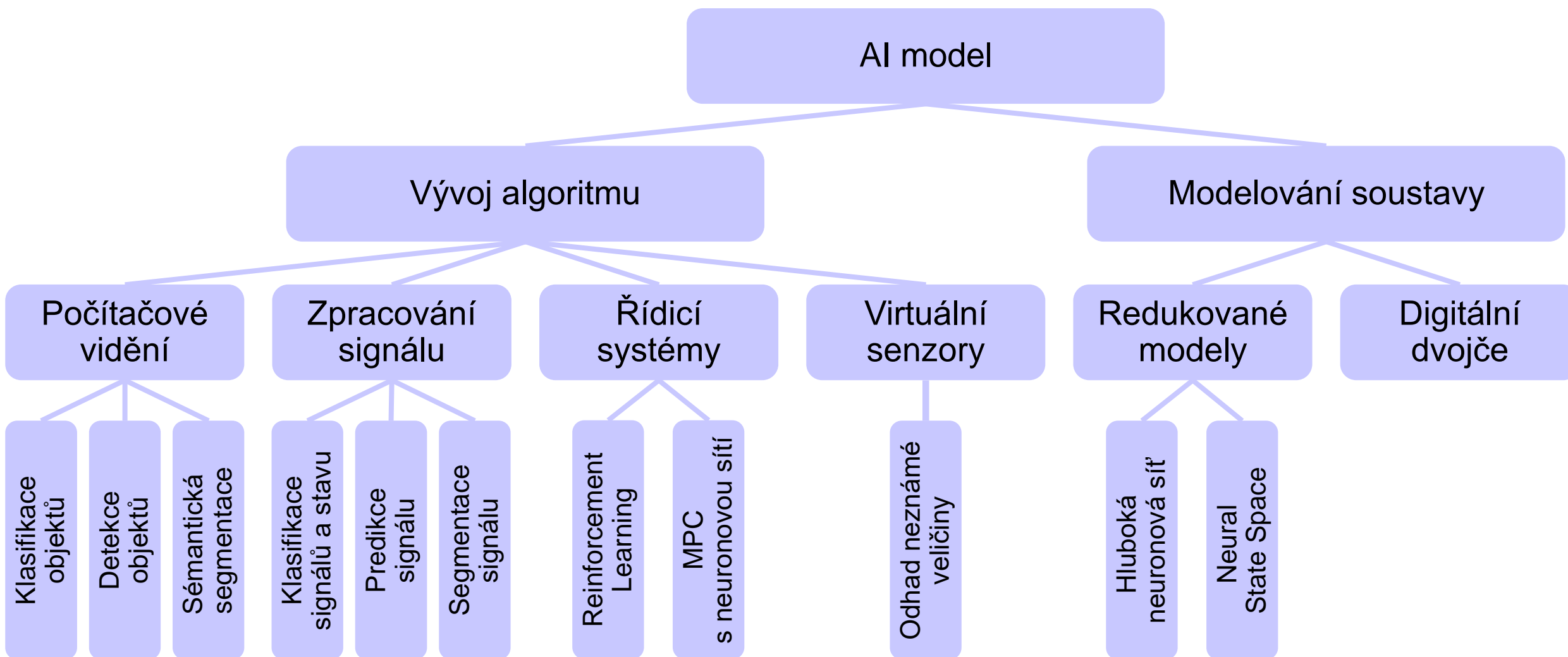


Využit předdefinované sítě a před-učené sítě

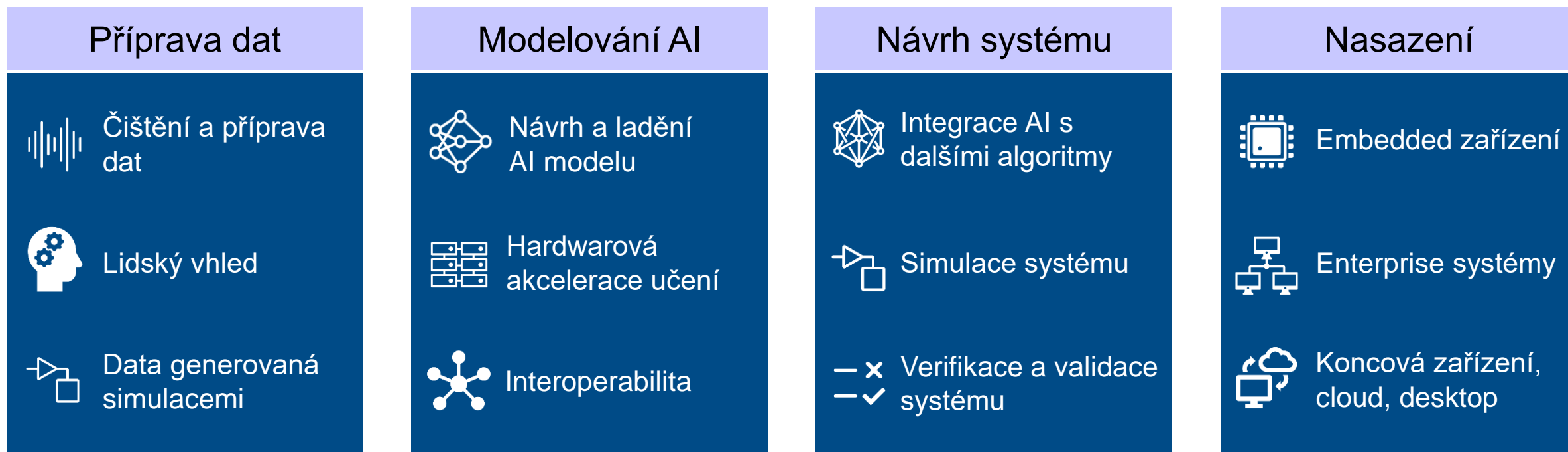
# AI a Model-Based Design



# Modelování AI – Typ algoritmu



# Postup návrhu systému využívajícího AI

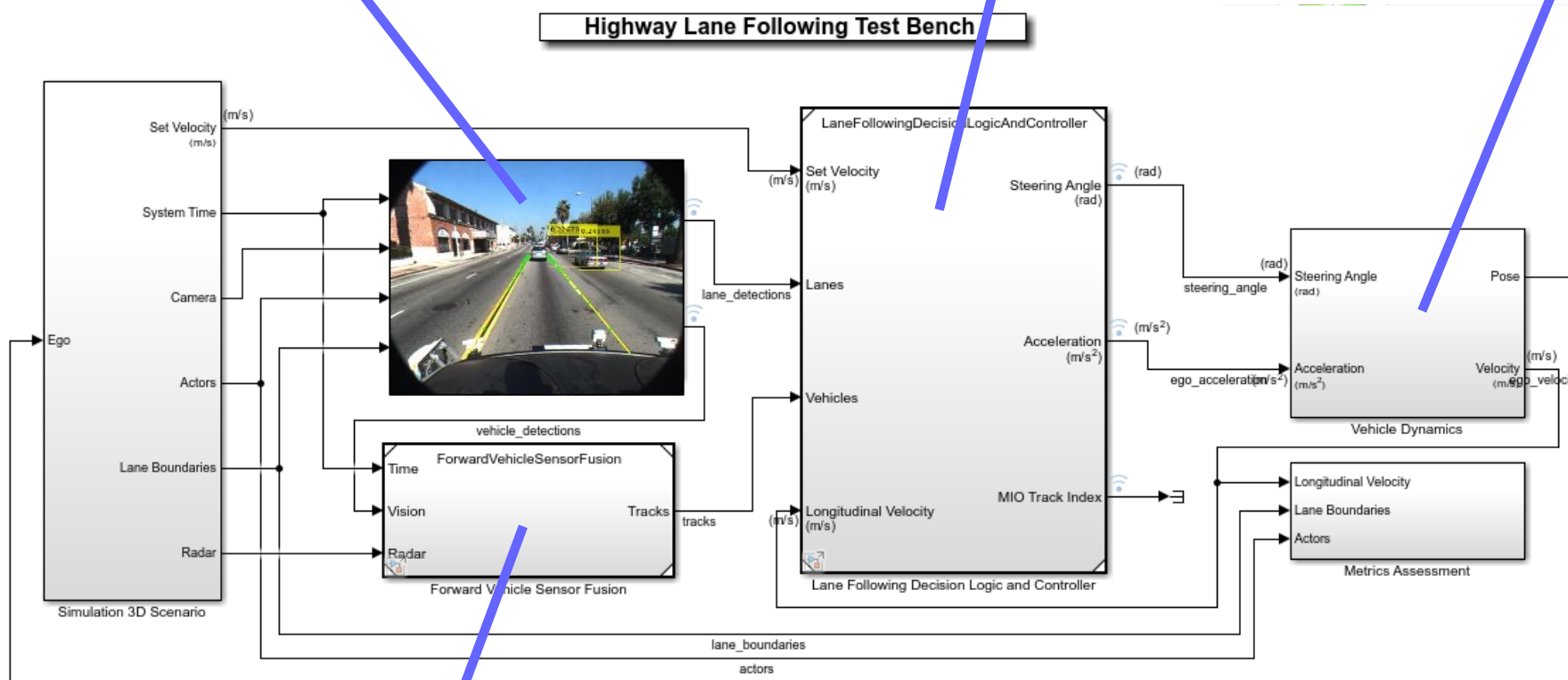


# Typický příklad: Asistenční systém automobilu

AI algoritmus  
detekce jízdních pruhů a vozidel

Řídicí systém

Model dynamiky vozidla  
pro testování chování systému



Senzorická fúze

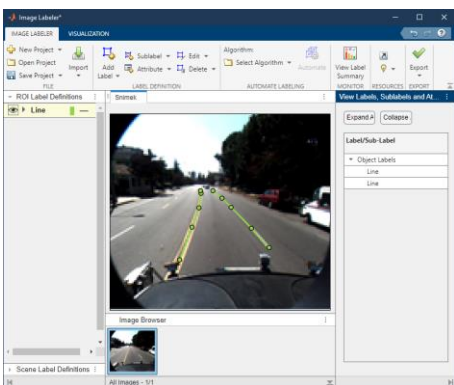
# Typický příklad: Asistenční systém automobilu

Příprava dat

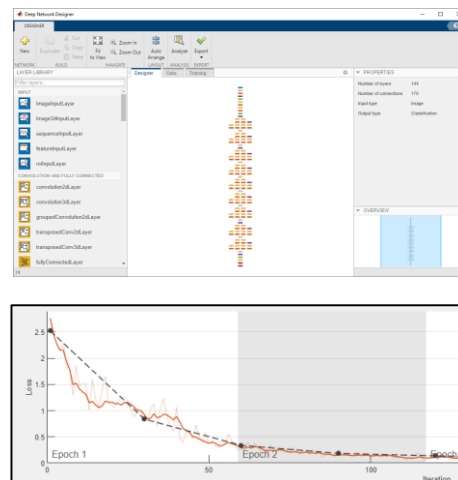
Modelování AI

Návrh systému

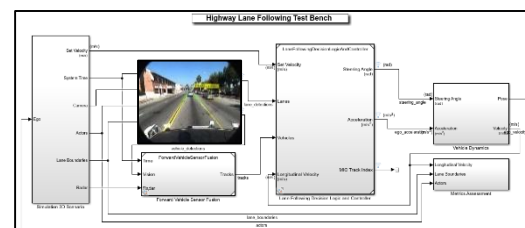
Nasazení



- úprava dat z kamery
- označení jízdních pruhů a vozidel v datech pro učení AI modelů



- výběr a úprava AI modelů (hluboké neuronové sítě)
- učení AI modelů
- ověření AI modelů



- propojení AI modelů s ostatními algoritmy
- simulace a testování modelu celého software

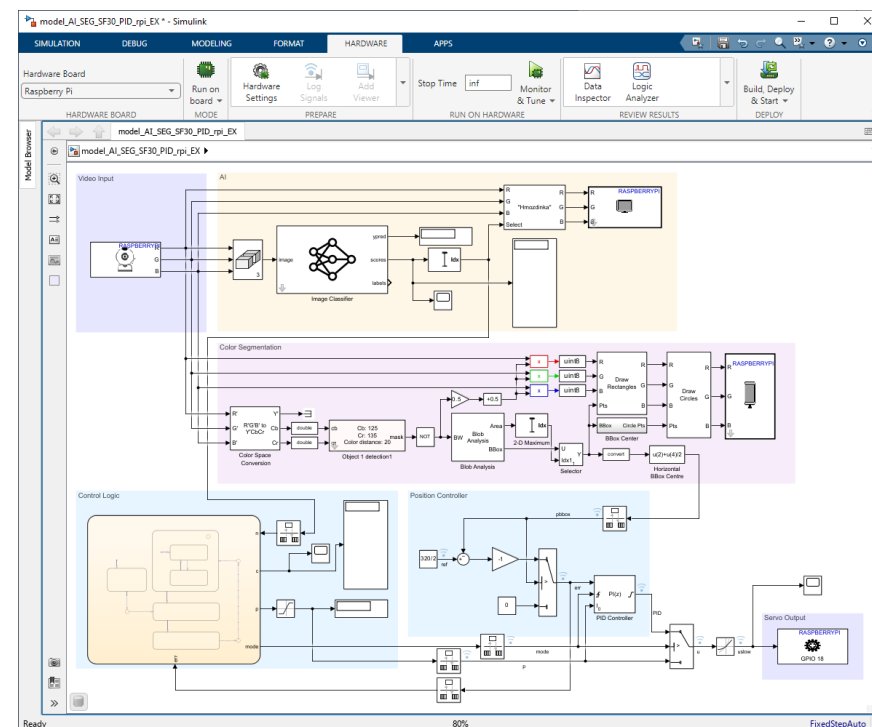


- generování kódu pro cílovou embedded platformu



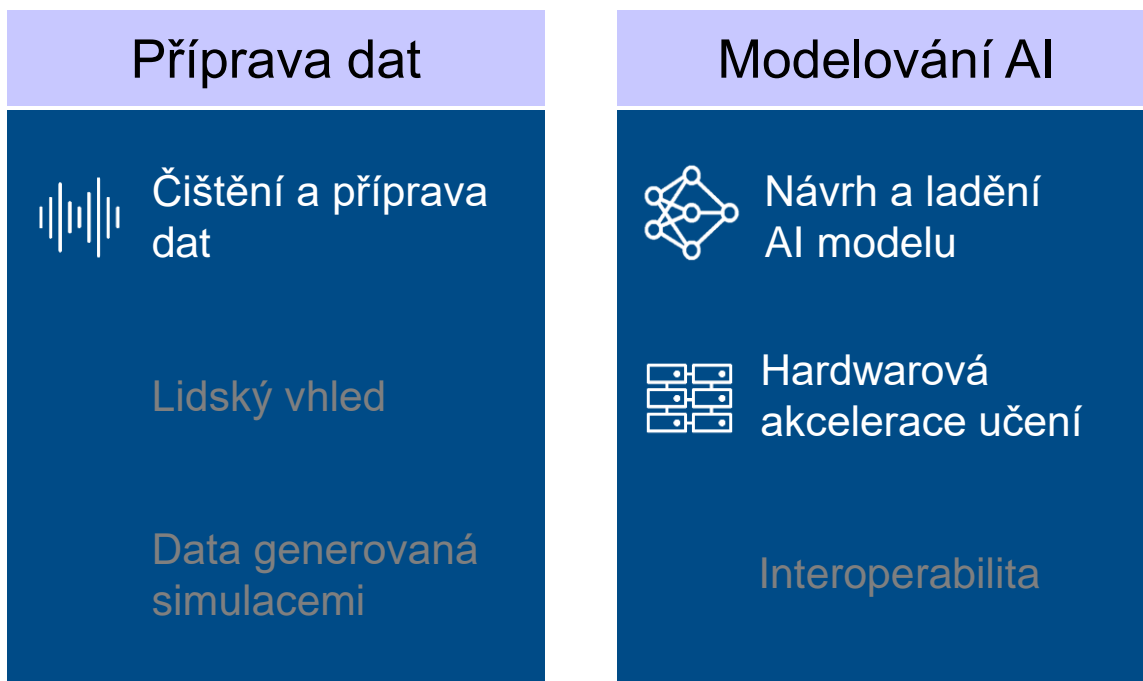
# Ukázka: Systém pro klasifikaci a počítání objektů

- Klasifikační algoritmus založený na AI
  - blok pro inferenci naučeného modelu z knihovny Deep Learning Toolbox
- Počítačové vidění
  - detekce polohy objektu prahováním
- Řídicí systém
  - vystředění polohy objektu pomocí PID regulace
- Přepínání režimů (Stateflow)
  - fixní natočení / doladění polohy / klasifikace objektu
- Hardware
  - Raspberry Pi 4, webkamera, servomotor

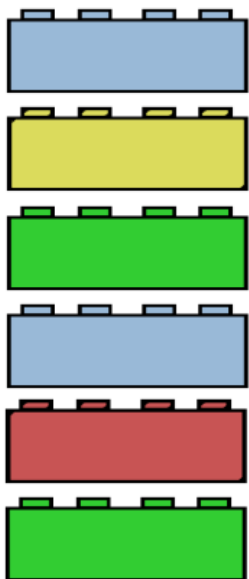


# Ukázka: Systém pro klasifikaci a počítání objektů

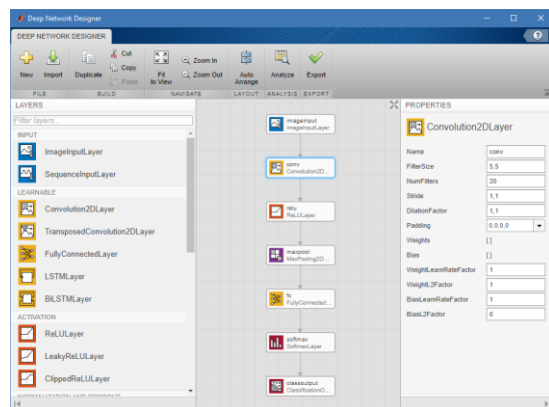
- 1. vytvoření AI modelu pro klasifikaci objektů z obrazových dat



# Vytvoření hluboké neuronové sítě v prostředí MATLAB



Deep Network Designer



učení  
v rámci  
aplikace

```
options = trainingOptions('sgdm');
net = trainNetwork(data, layers, opts);
```

```
results = classify(net, newData);
```

*vhodné pro většinu úloh*

Připravené funkce

```
layers = [imageInputLayer(inputSize)
convolution2dLayer(filterSize, numFilters)
reluLayer()
maxPooling2dLayer(poolSize)
fullyConnectedLayer(numClasses)
softmaxLayer()
classificationLayer()];
```

Přizpůsobení na míru

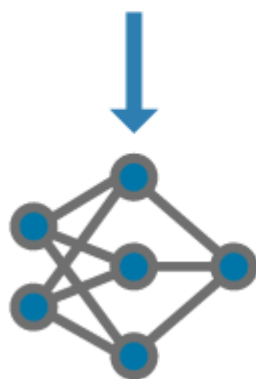
uživatelské smyčky pro učení sítí  
automatická diferenciace  
sdílené váhy  
uživatelské ztrátové funkce  
...

GAN, CGAN, siamské sítě, ...

# Transfer learning

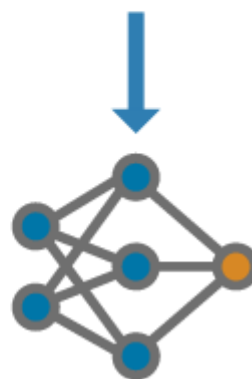
- Využití předdefinovaných a před-učených sítí

datová sada 1



model 1

datová sada 2

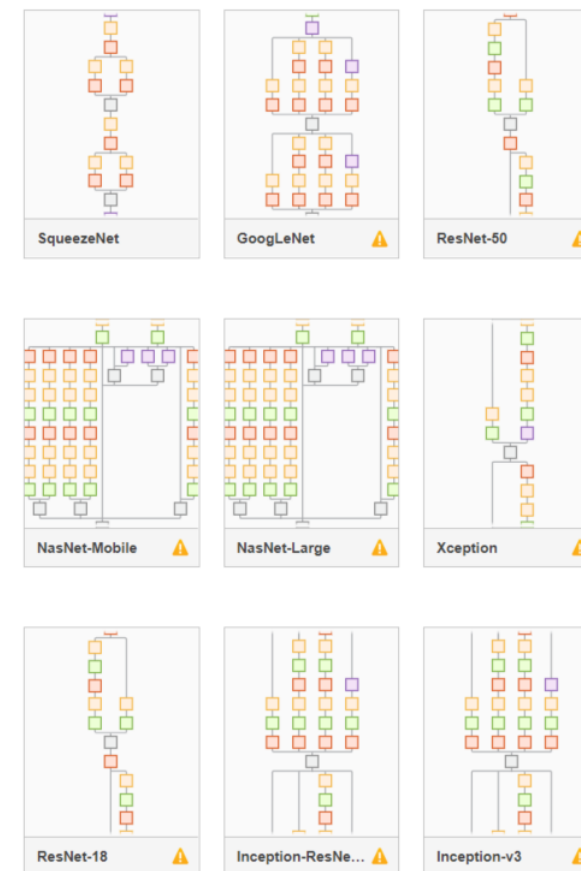


model 2

„znalost“ síť

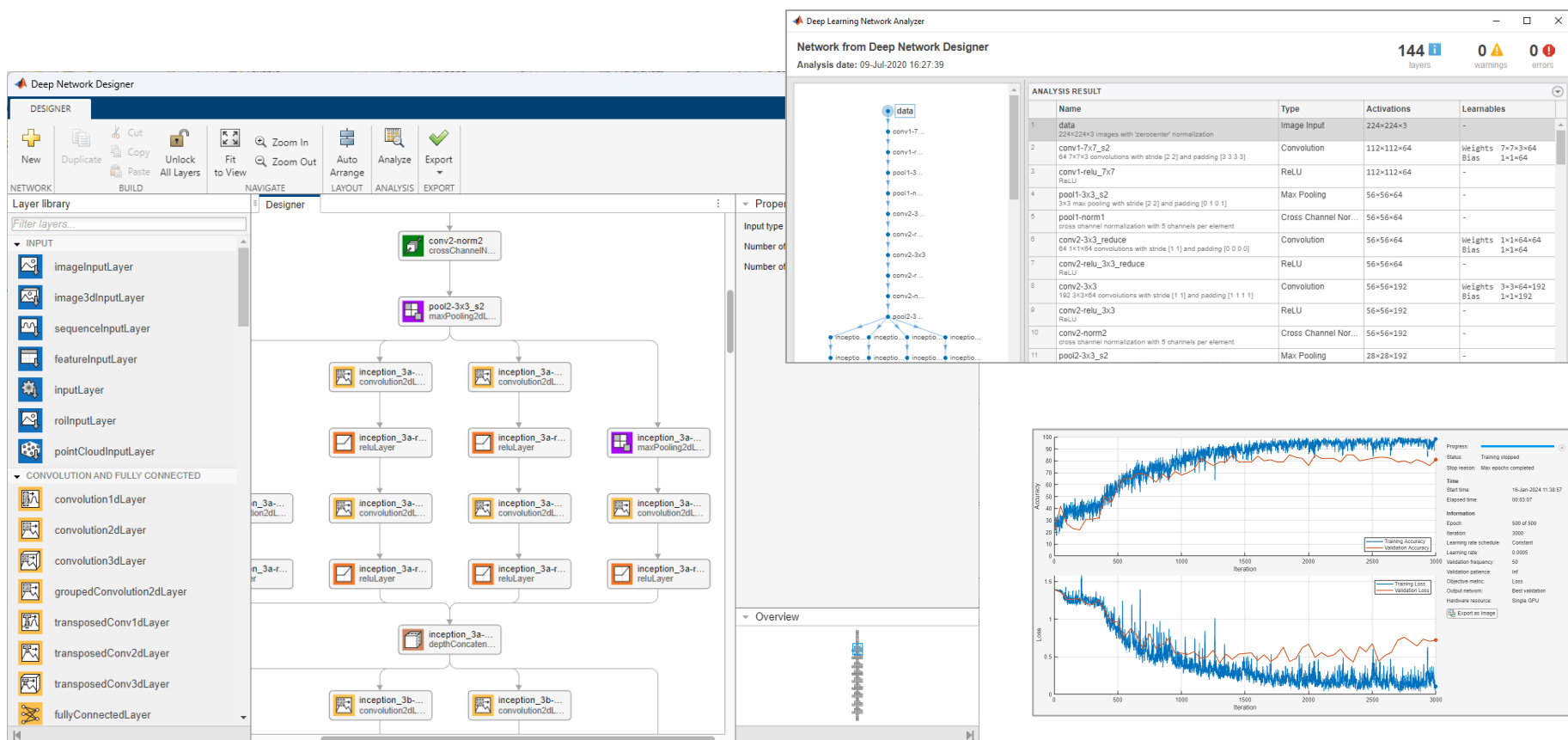


▼ Pretrained Networks



# Deep Network Designer

- Grafická tvorba a úprava sítí pro rychlejší návrh



The screenshot displays the Deep Network Designer interface, which includes a layer library, a central network design canvas, and an analysis window.

**Network Analysis Results:**

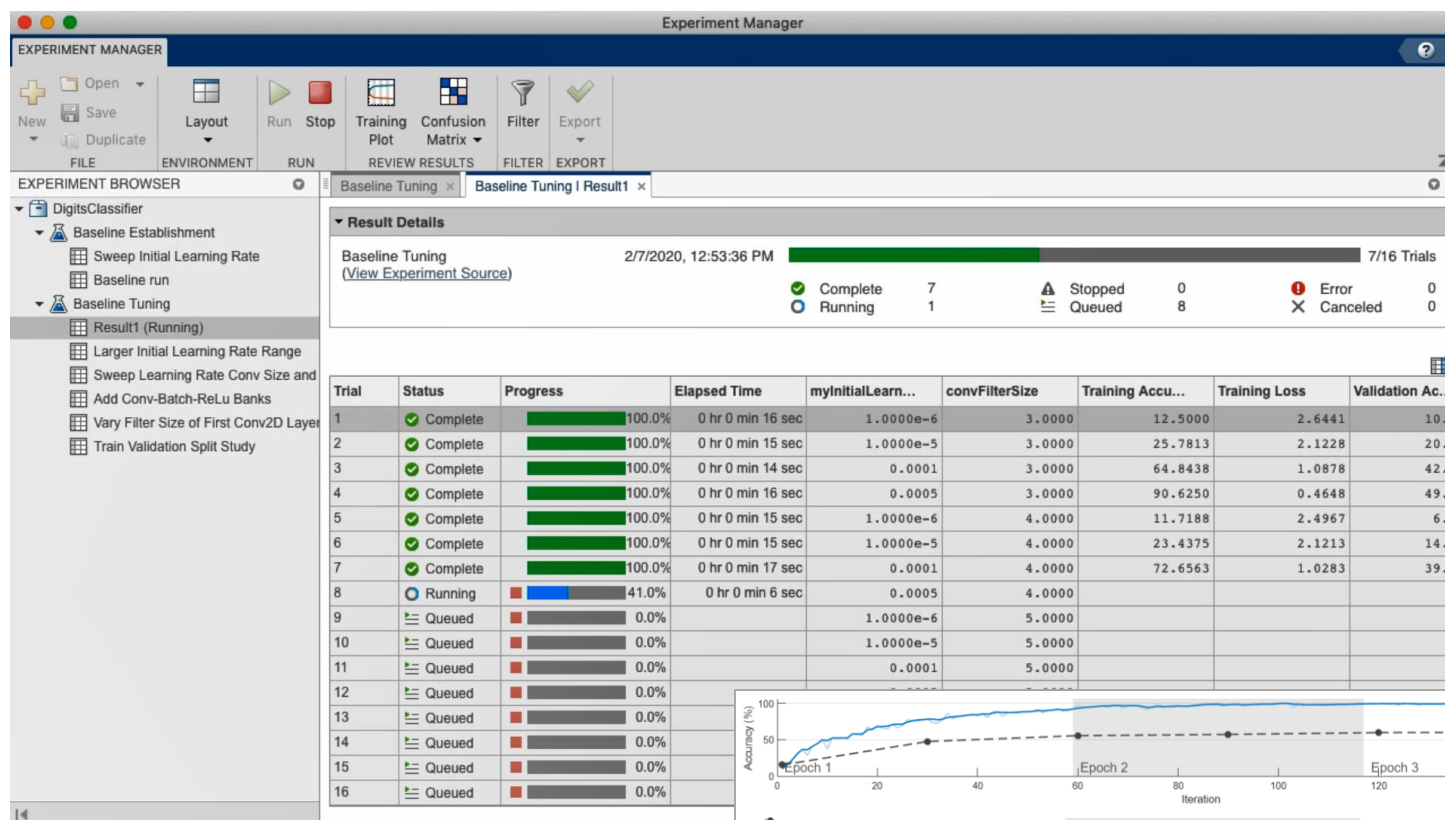
| Name   | Type                 | Activations | Learnables                         |
|--|----------------------|-------------|------------------------------------|
| 1 data<br>224x224x3 images with 'zerocenter' normalization                           | Image Input          | 224x224x3   | -                                  |
| 2 conv1-7x7_s2<br>64 7x7 convolutions with stride [2 2] and padding [3 3 3 3]        | Convolution          | 112x112x64  | Weights 7x7x3x64<br>Bias 1x1x64    |
| 3 conv1-relu_7x7<br>ReLU   | ReLU                 | 112x112x64  | -                                  |
| 4 pool1-3x3_s2<br>3x3 max pooling with stride [2 2] and padding [0 1 0 1]            | Max Pooling          | 56x56x64    | -                                  |
| 5 pool1-norm1<br>cross channel normalization with 5 channels per element             | Cross Channel Nor... | 56x56x64    | -                                  |
| 6 conv2-3x3_reduce<br>64 1x1x64 convolutions with stride [1 1] and padding [0 0 0 0] | Convolution          | 56x56x64    | Weights 1x1x64x64<br>Bias 1x1x64   |
| 7 conv2-relu_3x3_reduce<br>ReLU  | ReLU                 | 56x56x64    | -                                  |
| 8 conv2-3x3<br>192 3x3x64 convolutions with stride [1 1] and padding [1 1 1 1]       | Convolution          | 56x56x192   | Weights 3x3x64x192<br>Bias 1x1x192 |
| 9 conv2-relu_3x3<br>ReLU   | ReLU                 | 56x56x192   | -                                  |
| 10 conv2-norm2<br>cross channel normalization with 5 channels per element            | Cross Channel Nor... | 56x56x192   | -                                  |
| 11 pool2-3x3_s2  | Max Pooling          | 28x28x192   | -                                  |

**Training Progress:**

The training progress window shows Accuracy and Loss over 3000 iterations. Training Accuracy (blue line) increases from approximately 30% to 80%, while Validation Accuracy (orange line) increases from approximately 25% to 75%. Training Loss (blue line) decreases from approximately 1.5 to 0.5, and Validation Loss (orange line) decreases from approximately 1.0 to 0.5. The training process is complete.

# Experiment Manager

- Nalezení optimální sítě pomocí experimentů



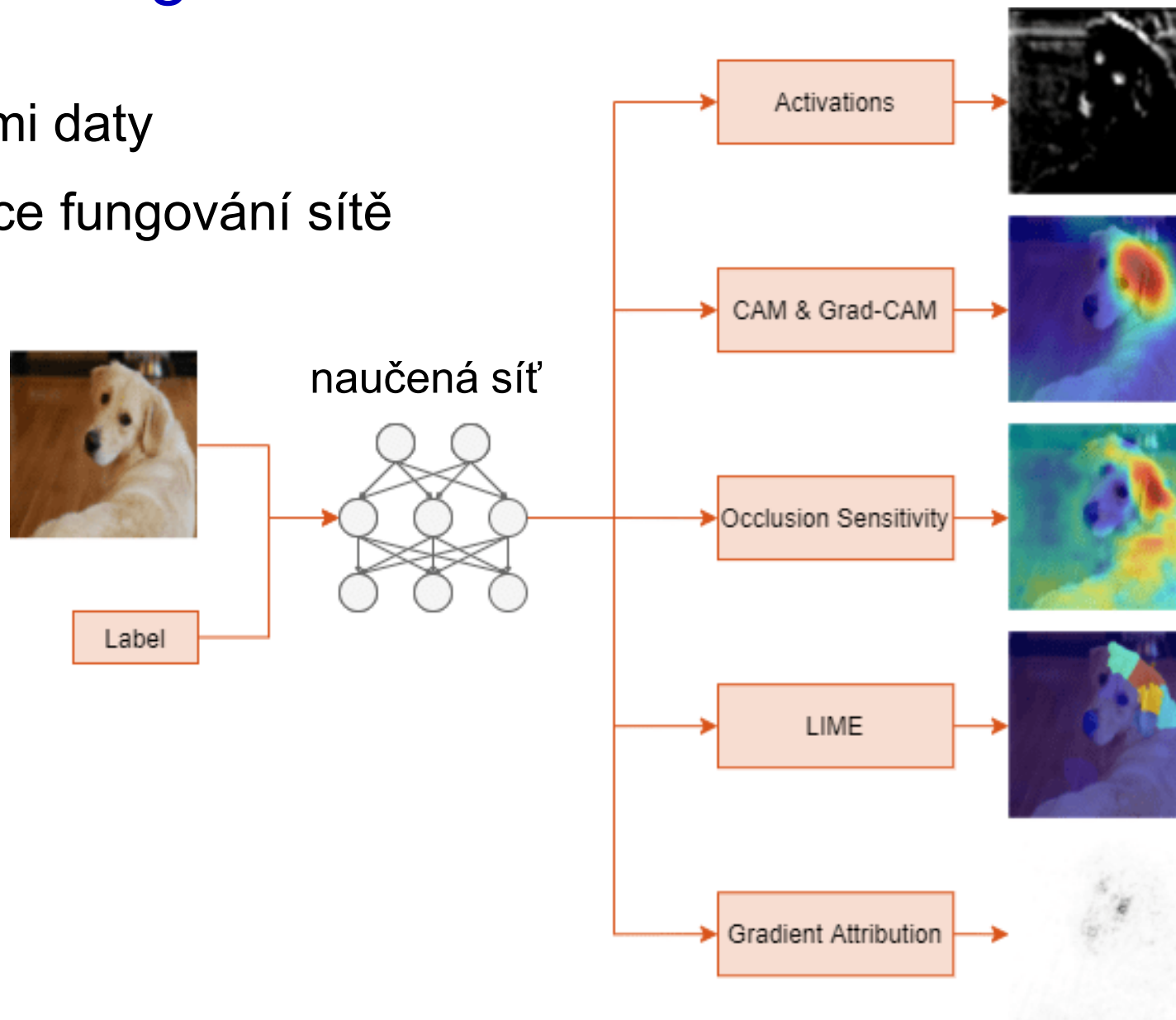
The screenshot displays the Experiment Manager interface. On the left is the 'EXPERIMENT BROWSER' showing a tree view of 'DigitsClassifier' with sub-items like 'Baseline Establishment', 'Baseline Tuning', and 'Result1 (Running)'. The main area shows 'Result Details' for 'Baseline Tuning' on 2/7/2020 at 12:53:36 PM, with 7/16 trials completed. A summary table shows 7 Complete, 1 Running, 8 Queued, 0 Stopped, 0 Error, and 0 Canceled trials.

| Trial | Status   | Progress | Elapsed Time      | myInitialLearn... | convFilterSize | Training Accu... | Training Loss | Validation Ac.. |
|-------|----------|----------|-------------------|-------------------|----------------|------------------|---------------|-----------------|
| 1     | Complete | 100.0%   | 0 hr 0 min 16 sec | 1.0000e-6         | 3.0000         | 12.5000          | 2.6441        | 10.             |
| 2     | Complete | 100.0%   | 0 hr 0 min 15 sec | 1.0000e-5         | 3.0000         | 25.7813          | 2.1228        | 20.             |
| 3     | Complete | 100.0%   | 0 hr 0 min 14 sec | 0.0001            | 3.0000         | 64.8438          | 1.0878        | 42.             |
| 4     | Complete | 100.0%   | 0 hr 0 min 16 sec | 0.0005            | 3.0000         | 90.6250          | 0.4648        | 49.             |
| 5     | Complete | 100.0%   | 0 hr 0 min 15 sec | 1.0000e-6         | 4.0000         | 11.7188          | 2.4967        | 6.              |
| 6     | Complete | 100.0%   | 0 hr 0 min 15 sec | 1.0000e-5         | 4.0000         | 23.4375          | 2.1213        | 14.             |
| 7     | Complete | 100.0%   | 0 hr 0 min 17 sec | 0.0001            | 4.0000         | 72.6563          | 1.0283        | 39.             |
| 8     | Running  | 41.0%    | 0 hr 0 min 6 sec  | 0.0005            | 4.0000         |                  |               |                 |
| 9     | Queued   | 0.0%     |                   | 1.0000e-6         | 5.0000         |                  |               |                 |
| 10    | Queued   | 0.0%     |                   | 1.0000e-5         | 5.0000         |                  |               |                 |
| 11    | Queued   | 0.0%     |                   | 0.0001            | 5.0000         |                  |               |                 |
| 12    | Queued   | 0.0%     |                   |                   |                |                  |               |                 |
| 13    | Queued   | 0.0%     |                   |                   |                |                  |               |                 |
| 14    | Queued   | 0.0%     |                   |                   |                |                  |               |                 |
| 15    | Queued   | 0.0%     |                   |                   |                |                  |               |                 |
| 16    | Queued   | 0.0%     |                   |                   |                |                  |               |                 |

Below the table is a graph showing 'Accuracy (%)' and 'Loss' over 'Iteration' (0 to 160). The top graph shows Accuracy increasing from ~50% to ~100% over 160 iterations. The bottom graph shows Loss decreasing from ~2.0 to ~0.5 over 160 iterations. The graph is divided into three epochs: Epoch 1 (0-60 iterations), Epoch 2 (60-120 iterations), and Epoch 3 (120-160 iterations).

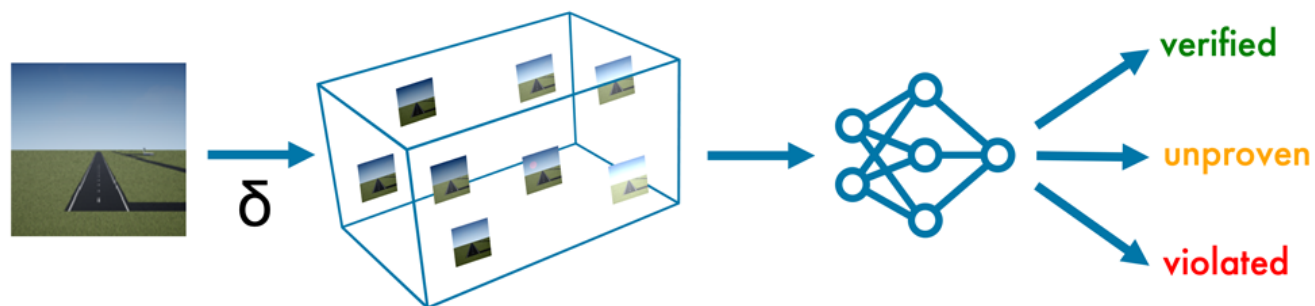
# Ověření správného fungování sítě

- Testování sítě s novými daty
- Vysvětlení a vizualizace fungování sítě



# Verifikace AI modelů

- Deep Learning Toolbox Verification Library
  - cíl: zajistit robustnost a spolehlivost hlubokých neuronových sítí
- Formální metody pro ověření robustnosti
  - proti tzv. adversarial examples



- Odhad citlivosti sítě na drobné změny vstupu
- Rozdělení dat na in-distribution a out-of-distribution
  - možnost sledování za běhu nasazené sítě



# Ukázka: Systém pro klasifikaci a počítání objektů

- 2. začlenění AI modelu do algoritmu pro ovládání stolu a počítání objektů

## Návrh systému



Integrace AI s  
dalšími algoritmy

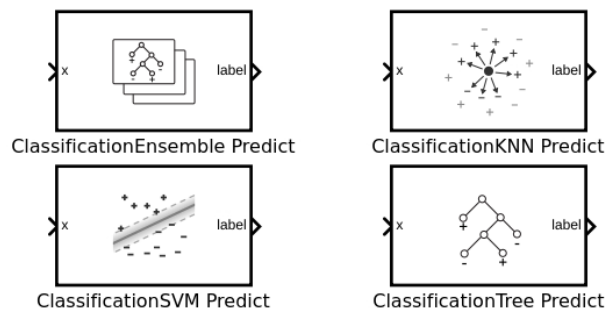
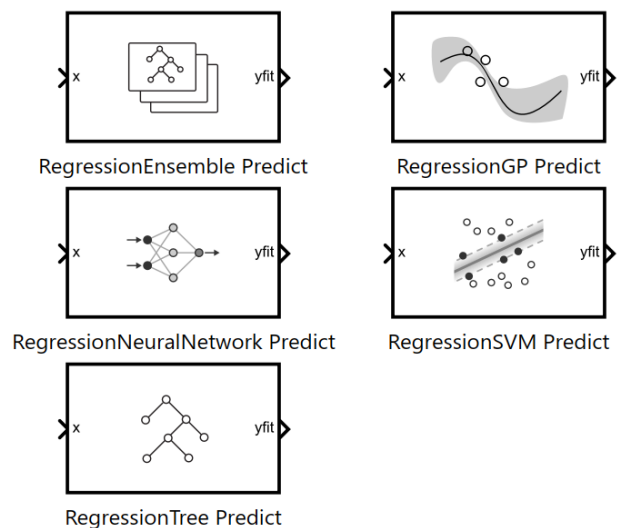


Simulace systému

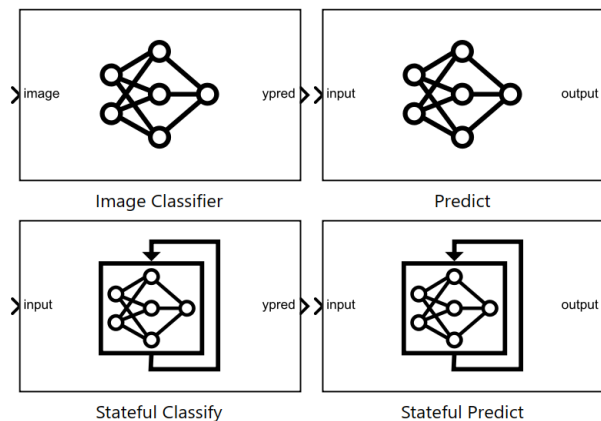


Verifikace a validace  
systému

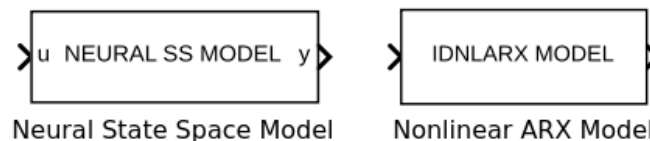
# Jak zahrnout AI model do Simulinku



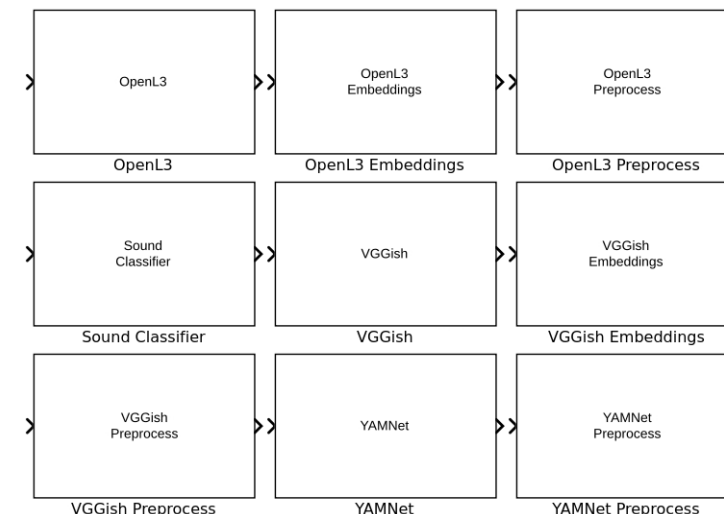
Statistics and Machine Learning Toolbox



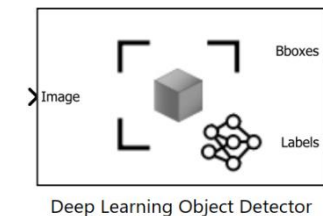
Deep Learning Toolbox



System Identification Toolbox



Audio Toolbox



Computer Vision Toolbox

## Ukázka: Systém pro klasifikaci a počítání objektů

- 3. nasazení algoritmu na cílovou platformu: automatické generování kódu

### Nasazení

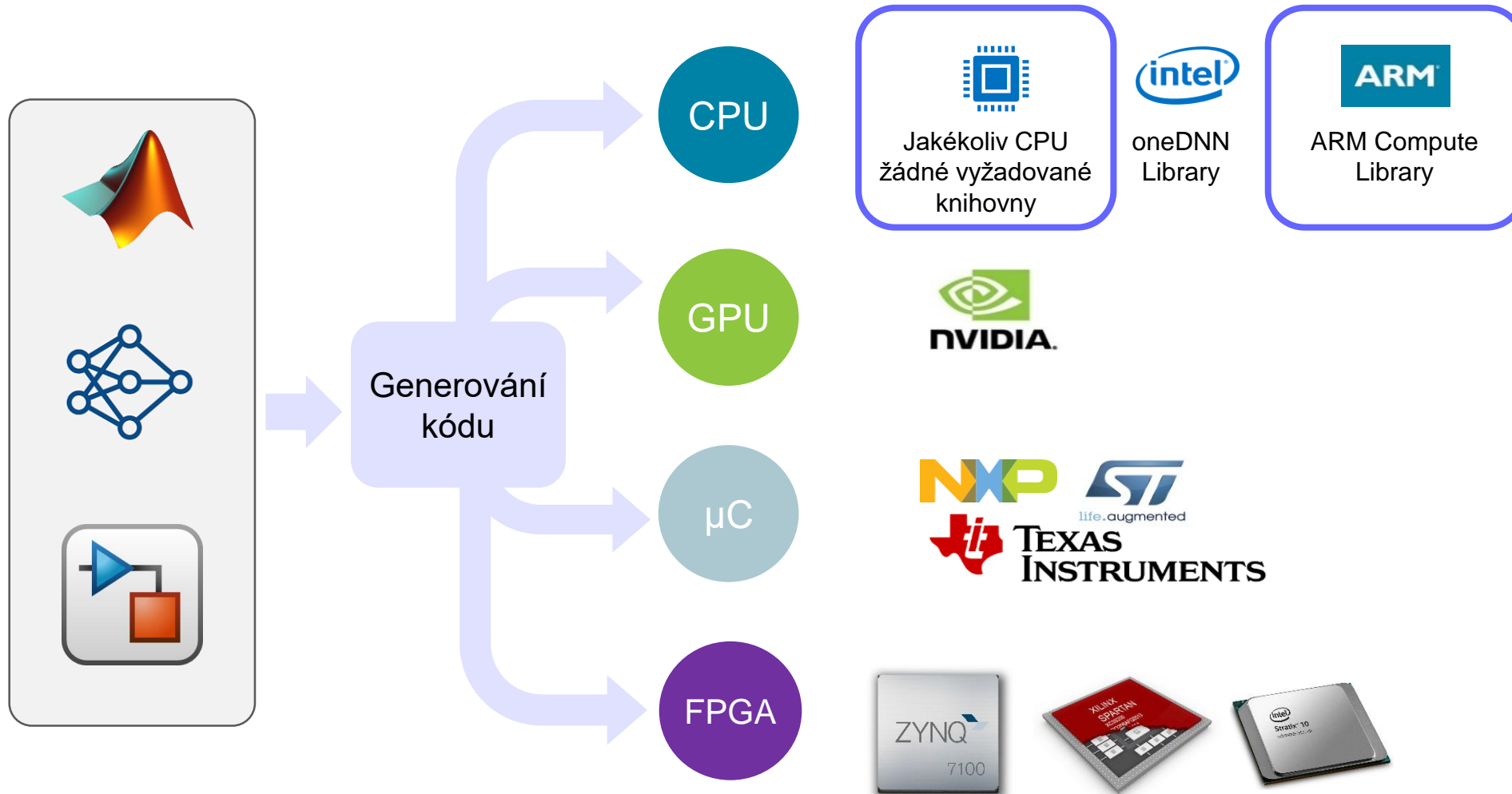


Embedded zařízení

Enterprise systémy

Koncová zařízení,  
cloud, desktop

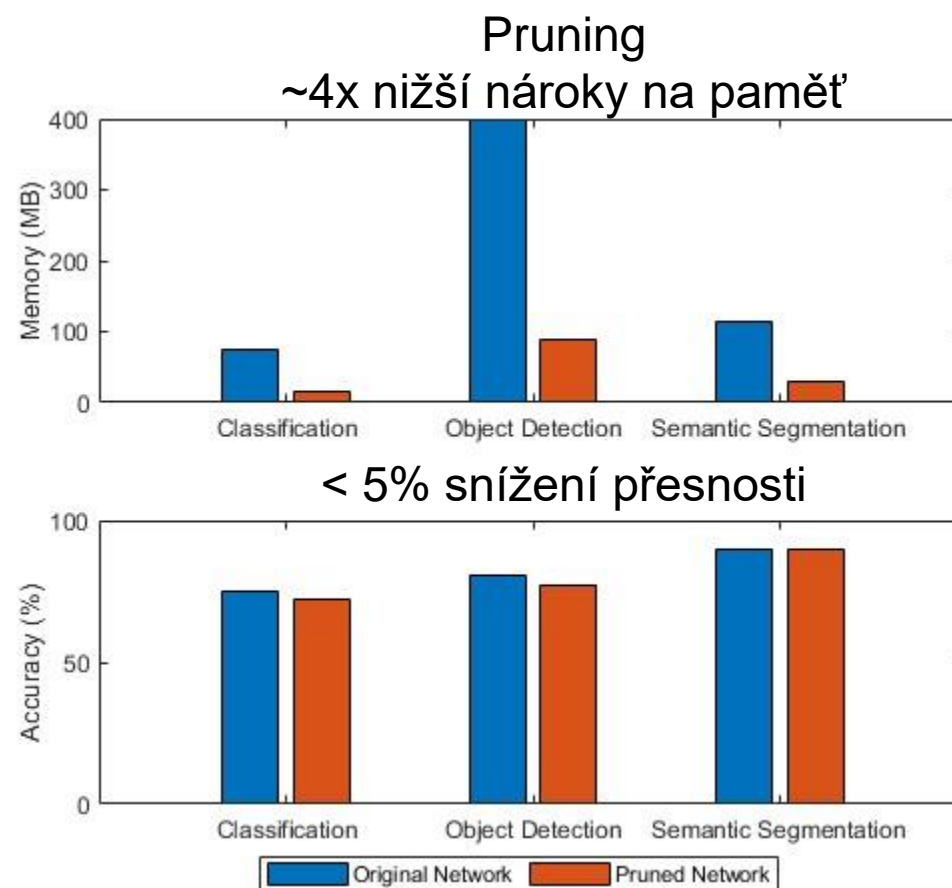
# Nasazení na cílovou platformu



# Redukce modelu

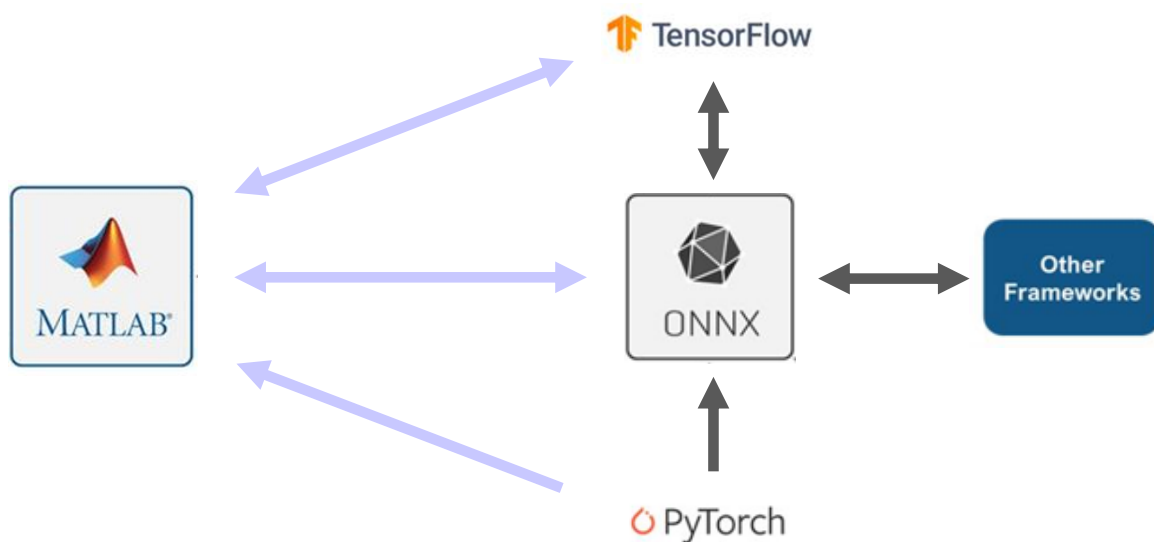
- Snížení paměťových a výpočetních nároků nasazeného modelu
- **Quantizace**
  - převod z floating point do fixed point aritmetiky
- **Pruning**
  - odstranění nedůležitých částí sítě

|  |        |
|--|--------|
| Deep Network Quantization App                  | R2020a |
| Taylor Pruning                                 | R2022a |
| LSTM Pruning                                   | R2022b |
| yolov3 and yolov4 object detector quantization | R2023a |



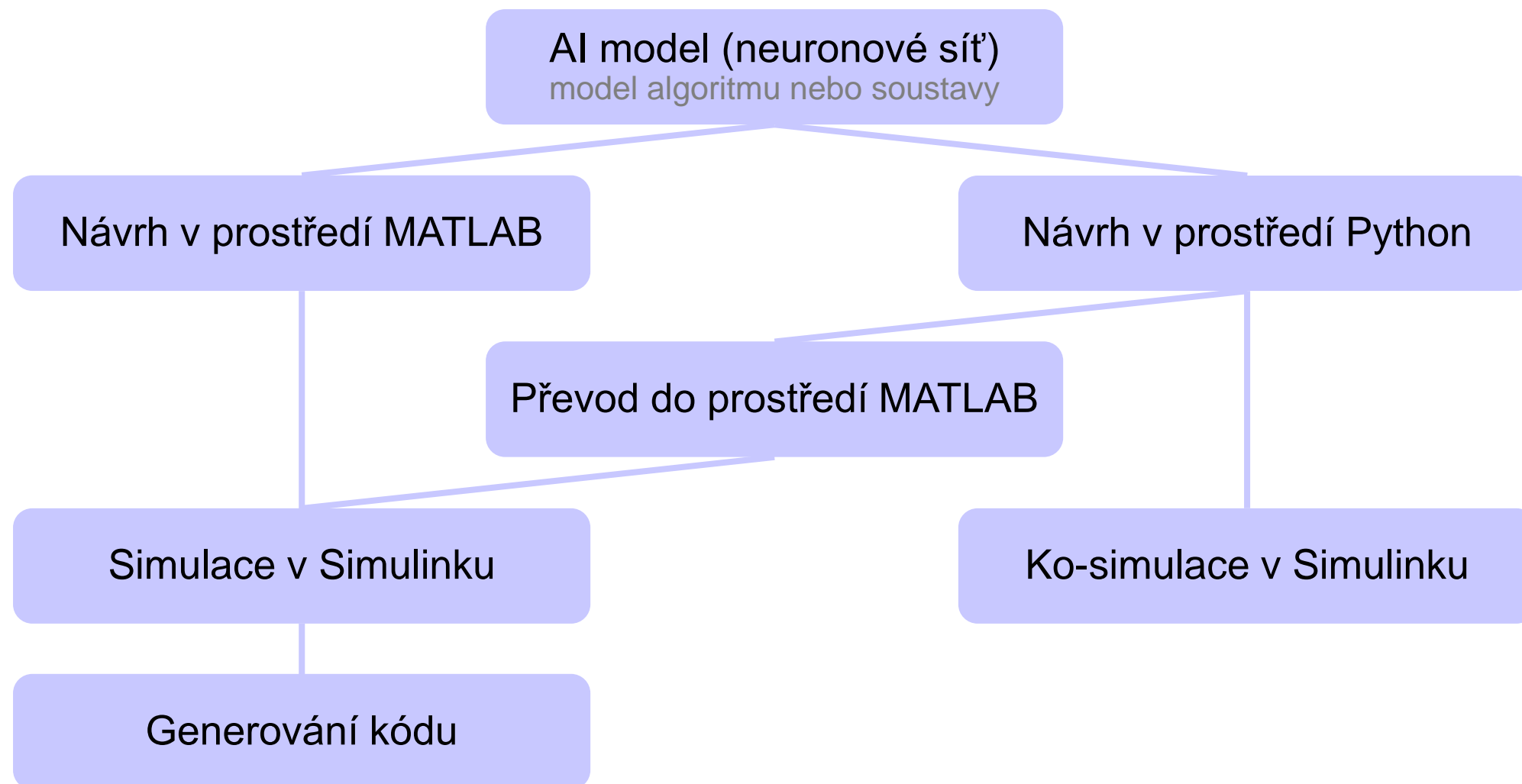
# MATLAB a spolupráce s dalšími nástroji

- Import modelů z prostředí TensorFlow, PyTorch
- Import modelů v otevřeném formátu ONNX



|   |               |
|---|---------------|
| TensorFlow-Keras Import   | <b>R2017b</b> |
| ONNX Converter (Import & Export)                                  | <b>R2018a</b> |
| TensorFlow Converter (Import)                                     | <b>R2021a</b> |
| TensorFlow Converter (Export)                                     | <b>R2022b</b> |
| PyTorch Converter (Import)  | <b>R2022b</b> |
| More layers, functions, operators, models, and versions supported | <b>R2023a</b> |

# Modelování AI – Postup tvorby modelu



# Podpora pro TensorFlow Lite

- Co je TFLite
  - open source nástroj pro deep learning
  - spouštění sítí na koncovém zařízení
- Simulace a nasazení
  - využití předučených modelů z TFLite

TFLite Support Package

**R2022a**

Support for Windows

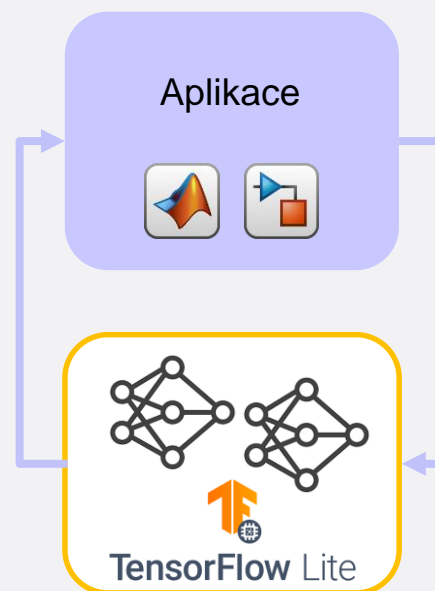
**R2022b**

Support for TFLite 2.8.0

**R2023a**

## Simulace

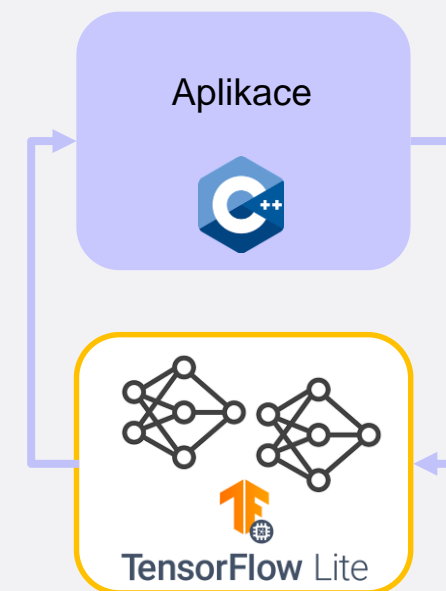
Aplikace spuštěná v  
MATLAB/Simulink



DL síť spuštěná v TFLite

## Generování kódu

Aplikace ve formě  
generovaného C++ kódu

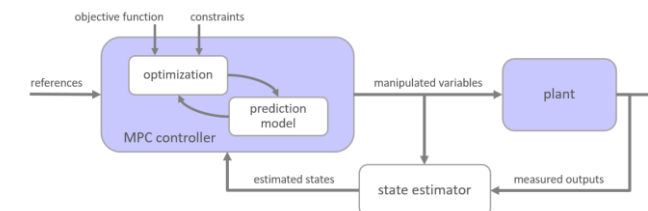
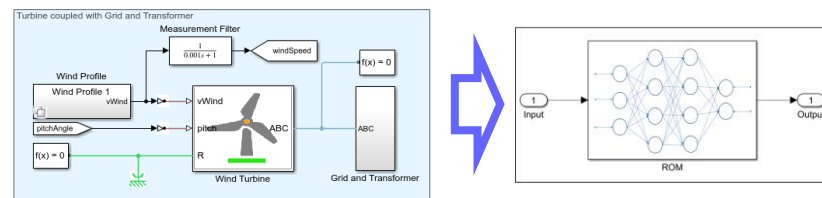
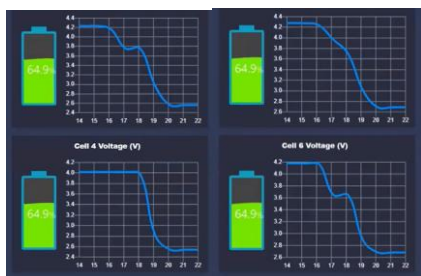


DL síť spuštěná v TFLite



# Další typické příklady společného použití AI a MBD

- Virtuální senzory
  - odhad veličiny, kterou nelze přímo měřit nebo je její měření neefektivní
- Redukované modely
  - rychlejší simulace pomocí zjednodušených modelů, které zachycují podstatné rysy chování
- Řídicí systémy
  - MPC s vnitřním predikčním modelem založeným na neuronových sítích
  - řídicí systém založený na metodě reinforcement learning



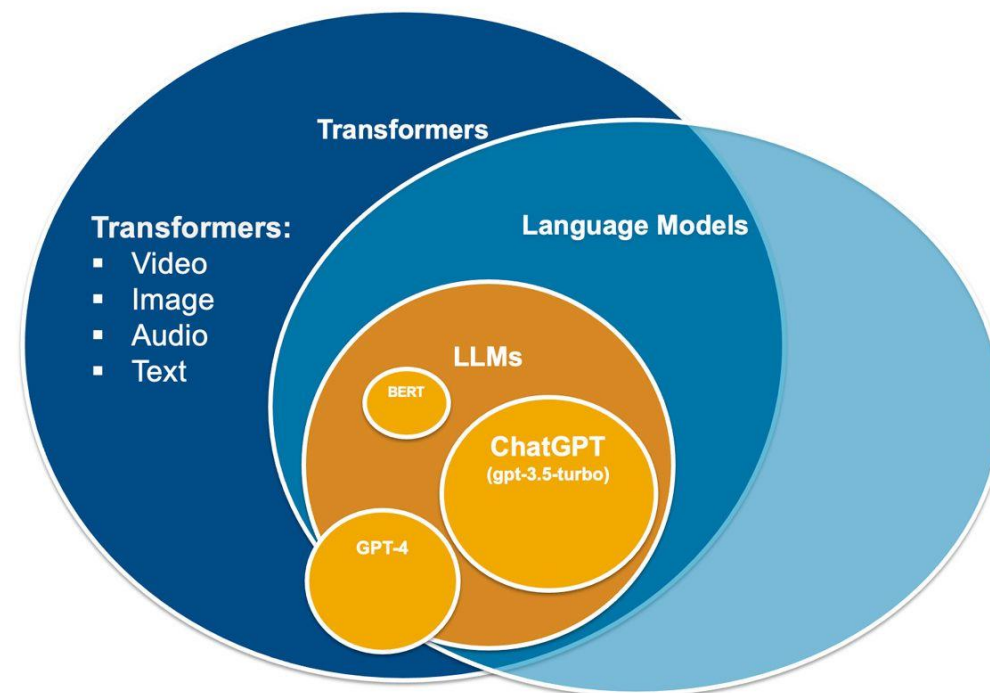
# AI v dalších toolboxech

- Computer Vision Toolbox
  - detektory objektů využívající deep learning (YOLO, R-CNN, SSD)
  - sítě pro sémantickou segmentaci
  - sítě pro detekci anomálií (pro automatickou vizuální inspekci)
- Lidar Toolbox
  - segmentace mračen bodů (point-cloud)
- Audio Toolbox
  - neuronové sítě a ML modely pro zpracování zvuku (YAMNet, VGGish, CREPE, i-vectors)
- System Identification Toolbox
  - modely typu neural state space (neural ODEs), nelineární ARX s nelinearitou ve formě AI
- Reinforcement Learning Toolbox

# GPT a BERT LLM modely

\* Generative Pretrained Transformer  
 \*\* Bidirectional Encoder Representations from Transformers

- Co je LLM (Large Language Model)
  - jazykový model na zpracování přirozeného jazyka
  - známé LLM modely: GPT\* a BERT\*\*
- ChatGPT
  - založen na technologii GPT-3.5 a GPT-4
  - <https://www.mathworks.com/discovery/chatgpt.html>
- LLM v prostředí MATLAB
  - k dispozici modely BERT, FinBERT a GPT-2 pro transfer learning
  - <https://github.com/matlab-deep-learning/transformer-models>
- AI Chat Playground
  - dotazy k MATLABu a generování programů MATLAB v jazyce MATLAB dle zadání
  - <https://www.mathworks.com/matlabcentral/playground>



Děkuji za pozornost