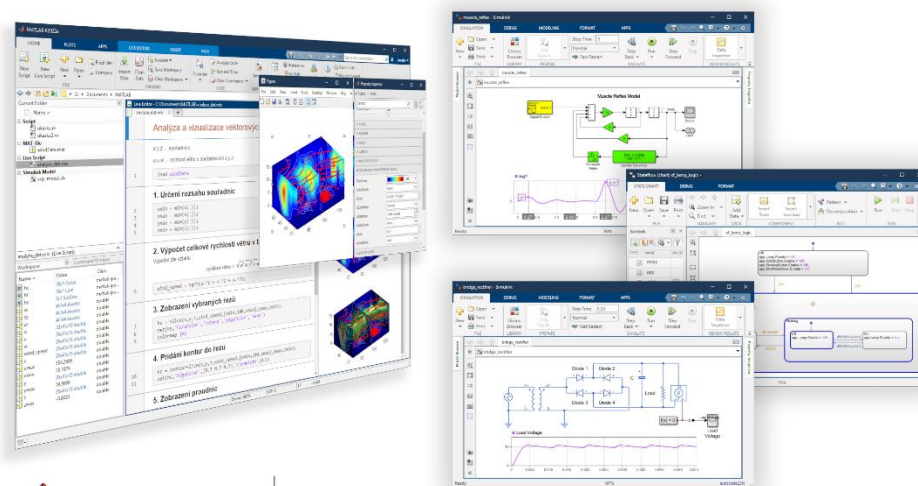


Master Class: MATLAB a Simulink pro vývoj systémů využívajících AI



Jaroslav Jirkovský
jirkovsky@humusoft.cz

www.humusoft.cz

info@humusoft.cz

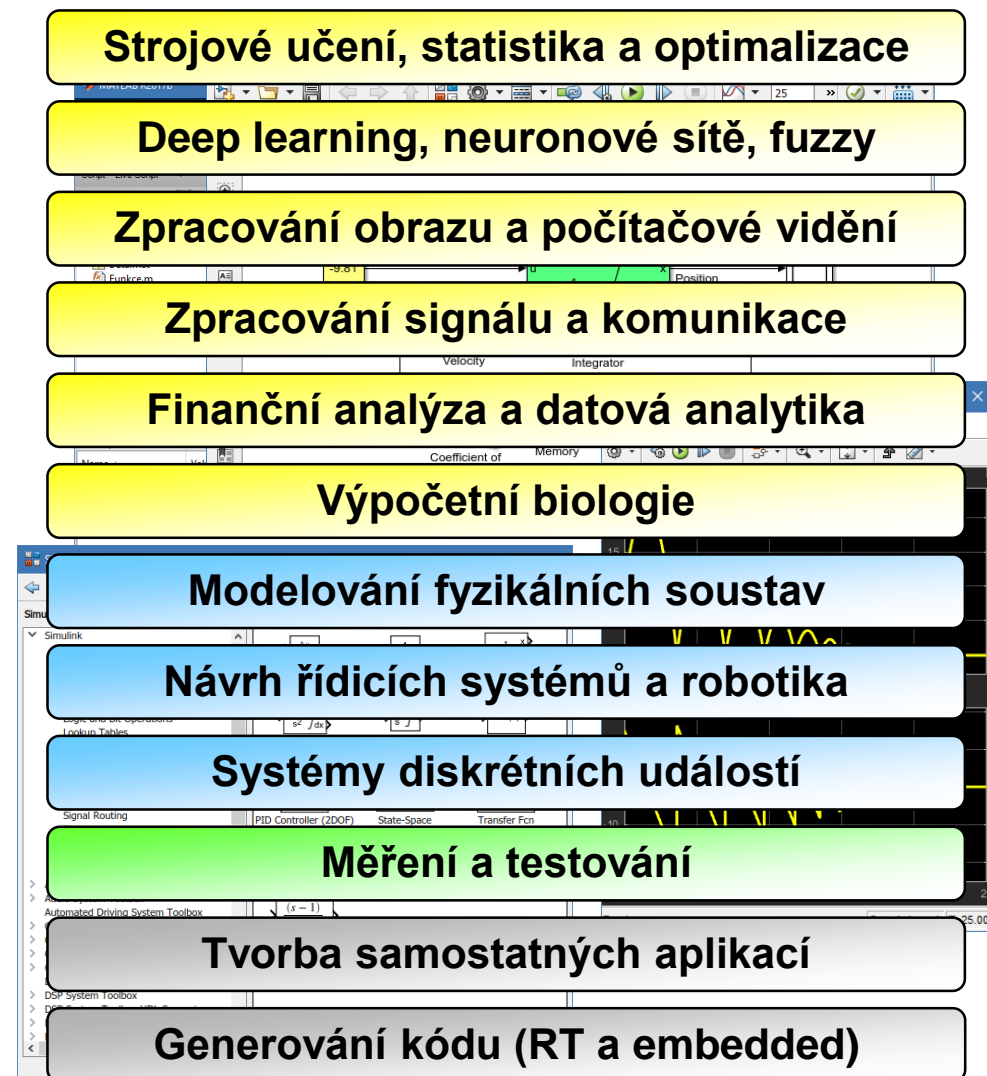
www.mathworks.com

Obsah

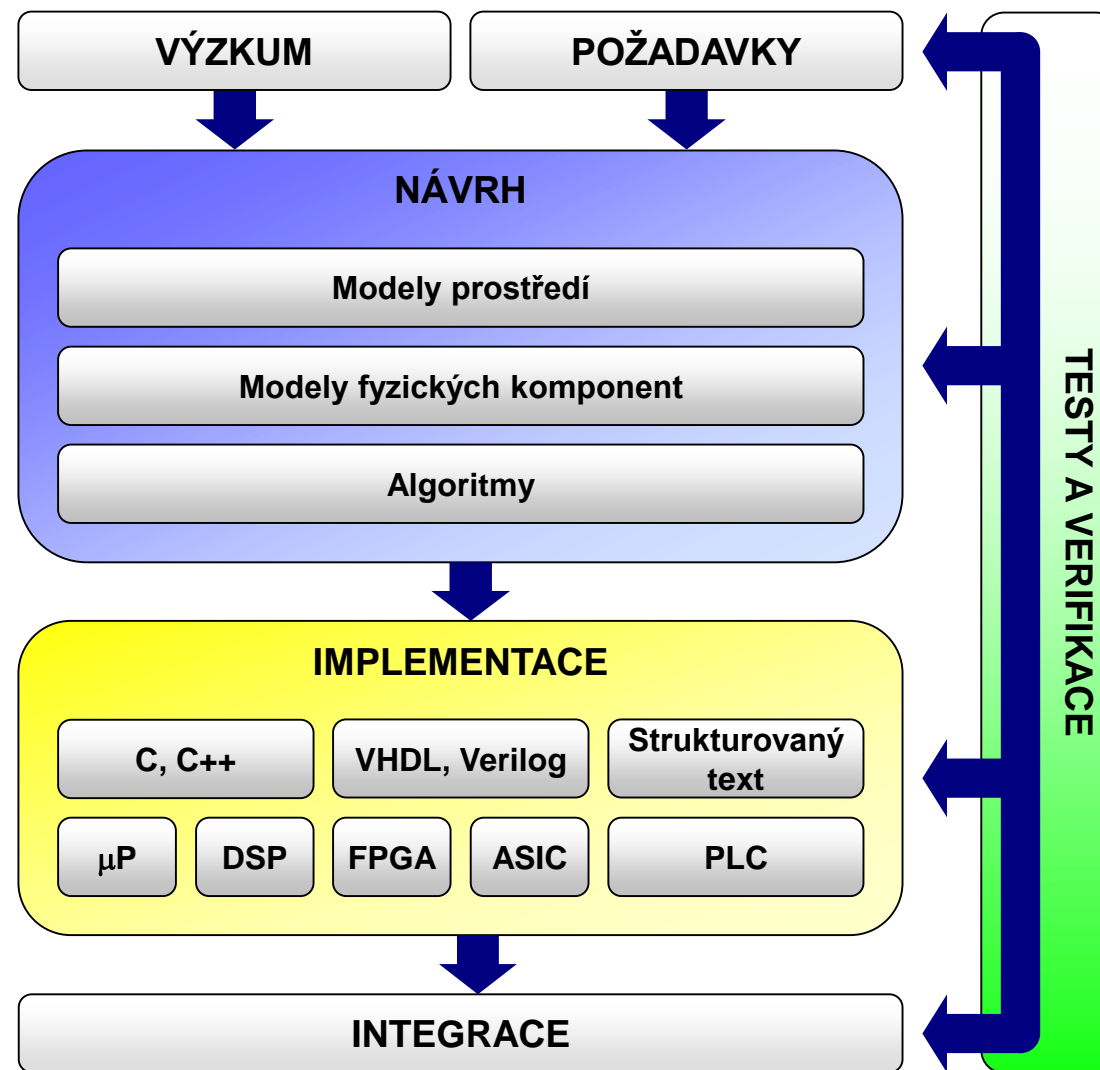
- Metoda Model-Based Design
 - co je metoda Model-Based Design
- Machine learning a deep learning
 - jak vytvořit a naučit model založený na umělé inteligenci
 - integrace modelu s ostatními algoritmy
- Nasazení algoritmů na embedded platformy
 - přizpůsobení modelu umělé inteligence požadavkům embedded platforem
 - automatické generování kódu pro embedded hardware

Co je MATLAB a Simulink

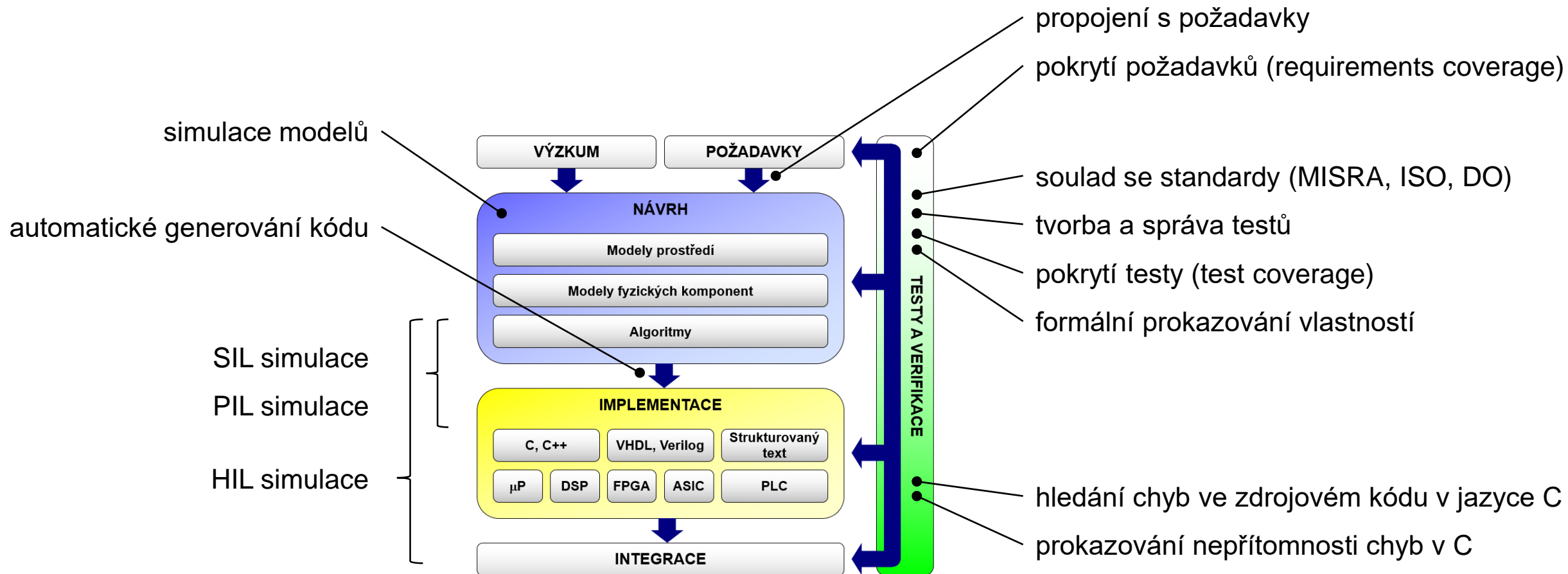
- **MATLAB**
 - inženýrský nástroj a interaktivní prostředí pro vědecké a technické výpočty
 - grafické a výpočetní nástroje
 - grafické aplikace (APPS)
 - otevřený systém
- **Simulink**
 - nadstavba MATLABu
 - modelování, simulace a analýza dynamických systémů
 - prostředí blokových schémat
 - platforma pro Model Based Design
- **Aplikační knihovny**



Vývoj metodou Model-Based Design




Vývoj metodou Model-Based Design



System

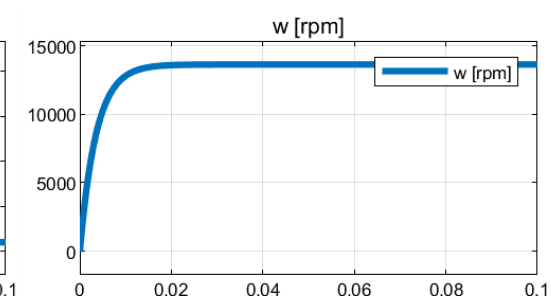
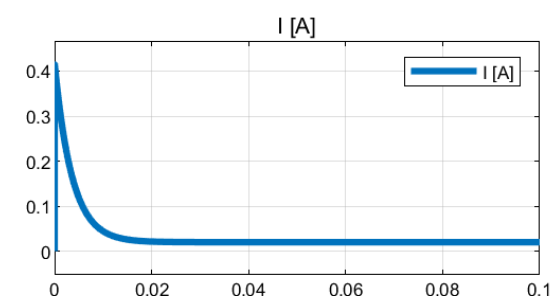
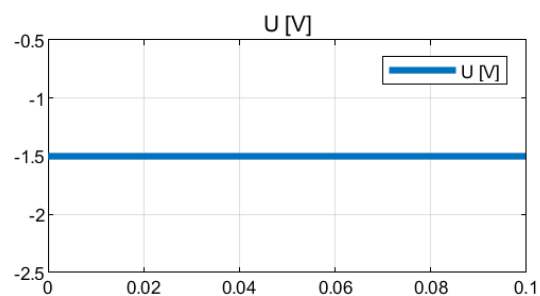
SYSTEM

vstup

 napětí U

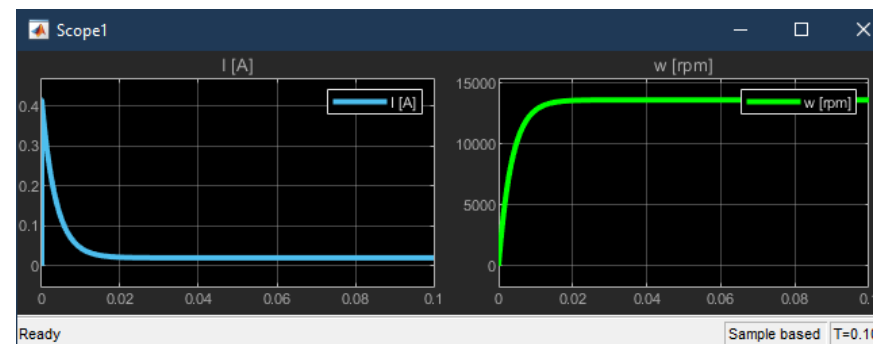
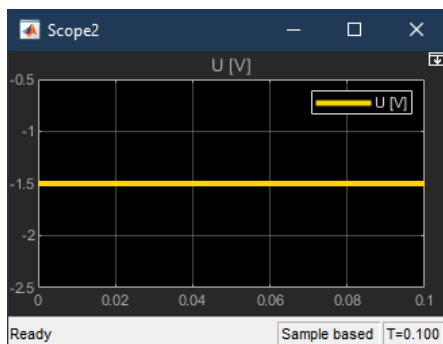
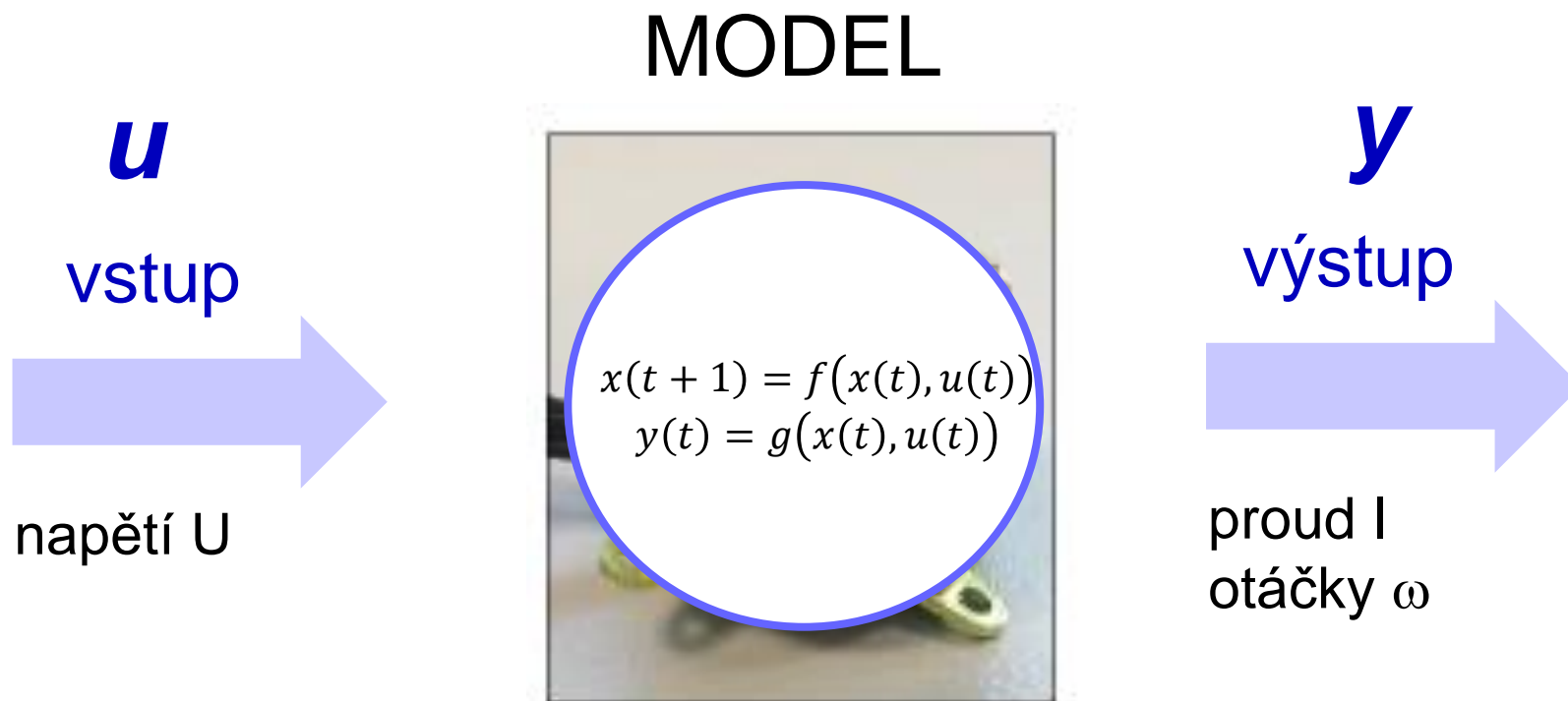


výstup

 proud I
 otáčky ω



Matematický model systému

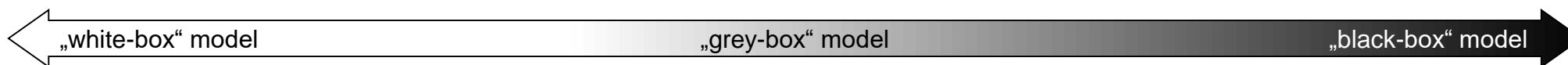


Přístupy k modelování

- Pro různé situace jsou vhodné různé přístupy

Fyzikální vztahy

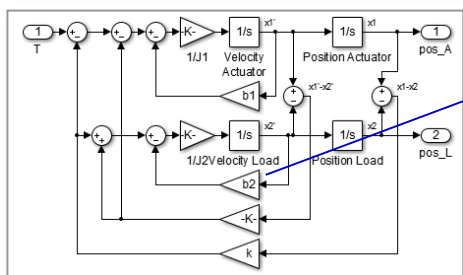
Naměřená data



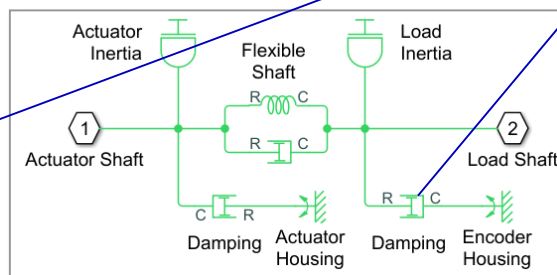
Modelování rovnic

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

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

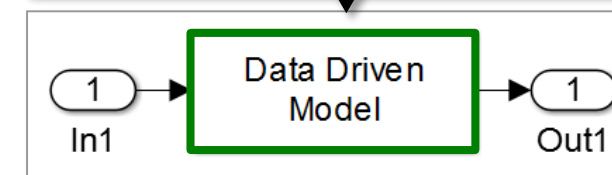
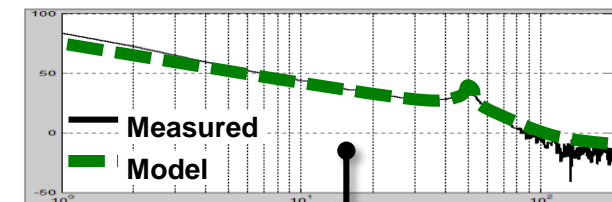


Fyzikální síť



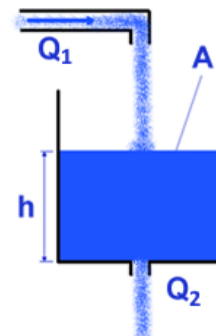
Ladění neznámých parametrů

Identifikace soustav

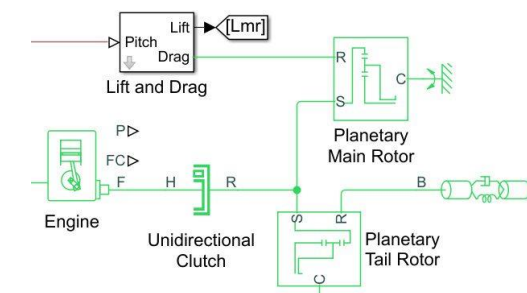
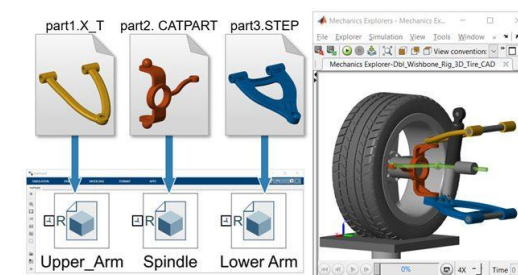
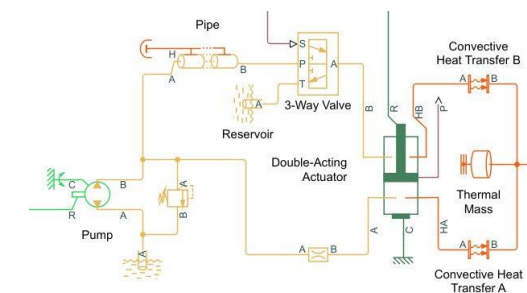
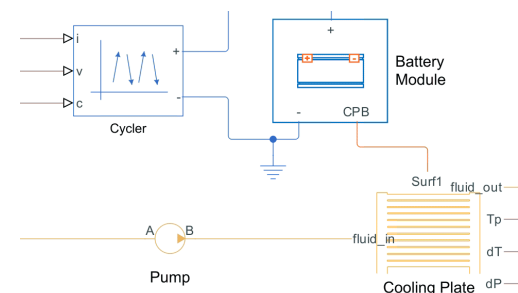
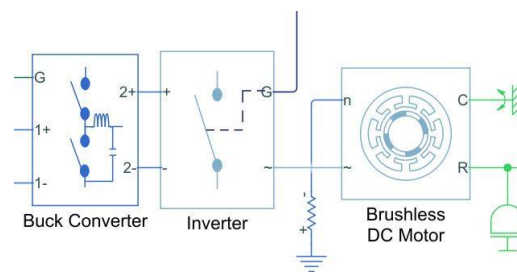
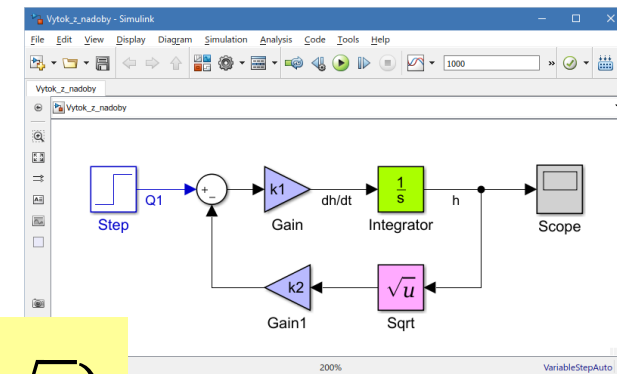


Modelování soustav

- 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



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

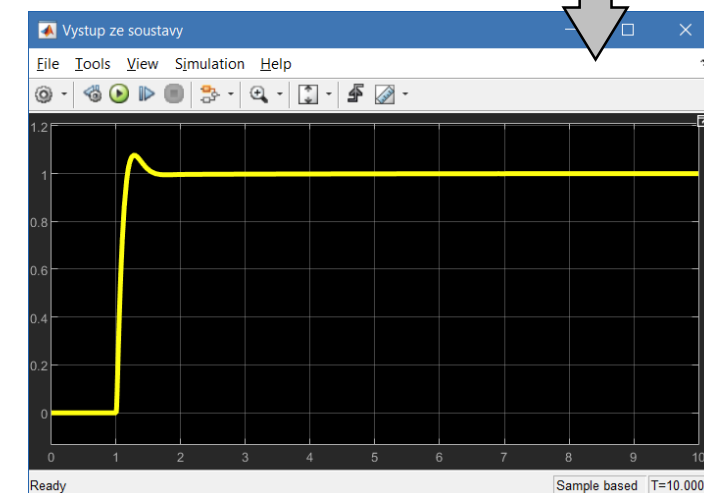
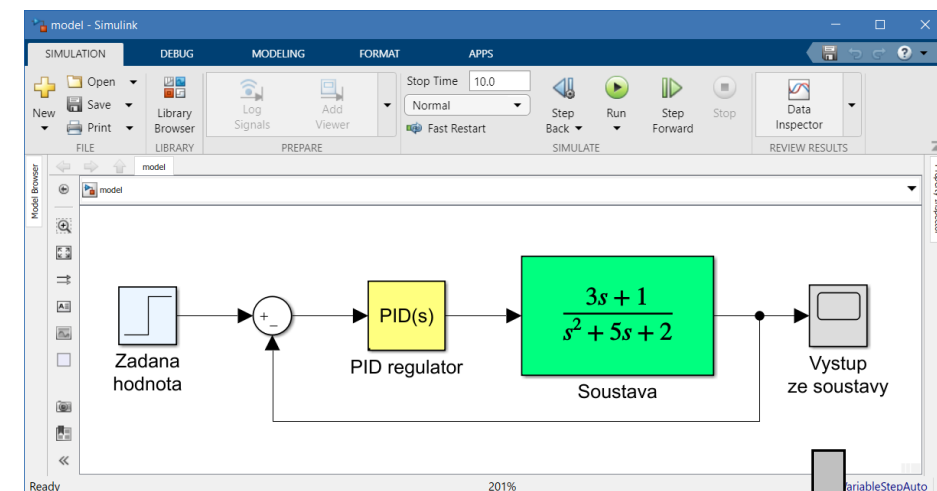


Modelování algoritmů

- 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



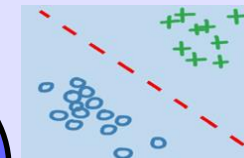
MACHINE LEARNING

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

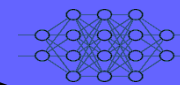
UNSUPERVISED LEARNING
(neoznačená data)



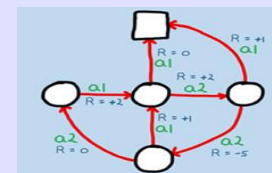
SUPERVISED LEARNING
(označená data)



DEEP LEARNING
(Neuronové sítě s mnoha vrstvami)



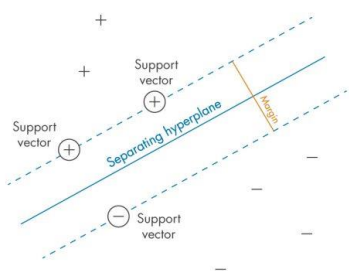
REINFORCEMENT LEARNING
(data z interakce)



AI modely v prostředí MATLAB

Machine learning

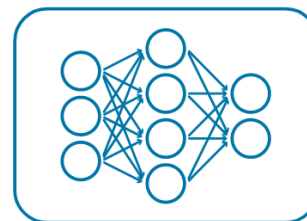
Deep learning



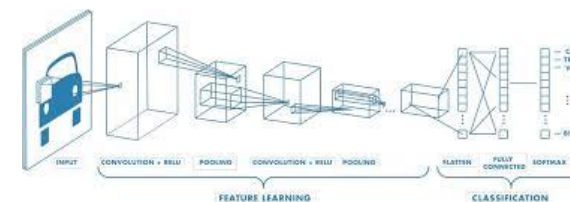
SVM



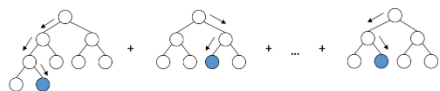
Clustering



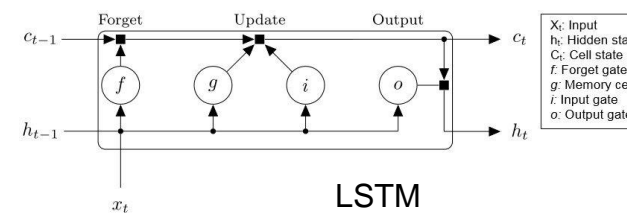
FC



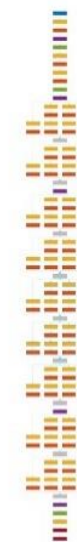
CNN



Decision trees



LSTM



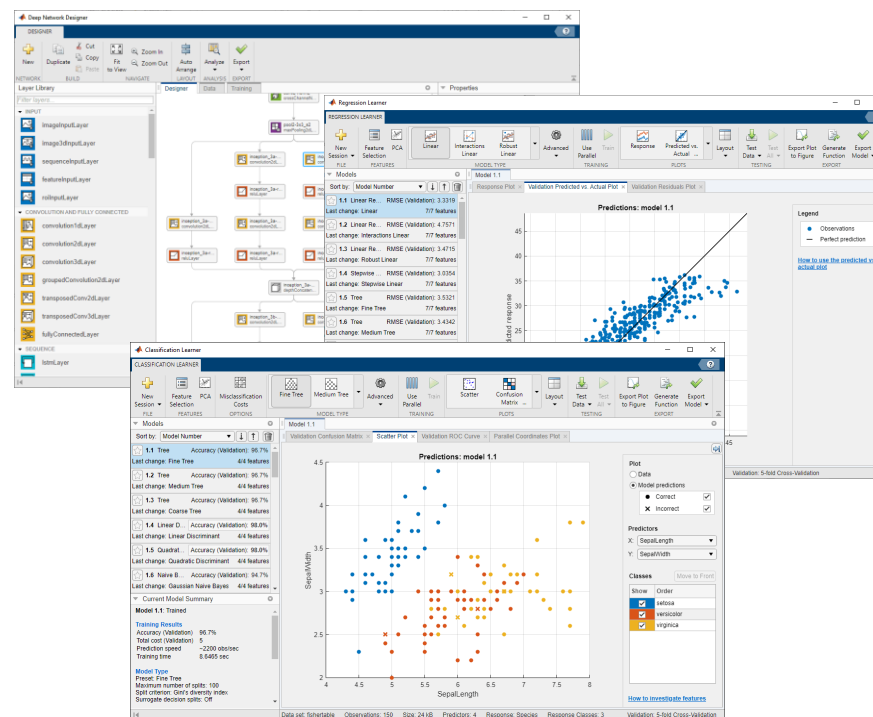
3 cesty k vytvoření AI modelu v prostředí MATLAB

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

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

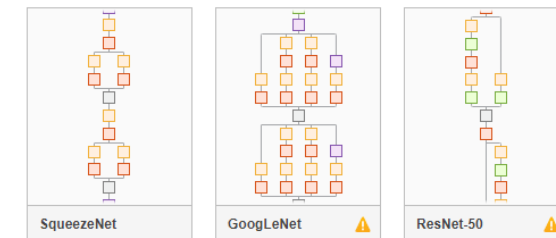
fitcauto / fitrauto

Funkce, skripty



Interaktivní návrh prostřednictvím grafických aplikací

Image Networks (Pretrained)



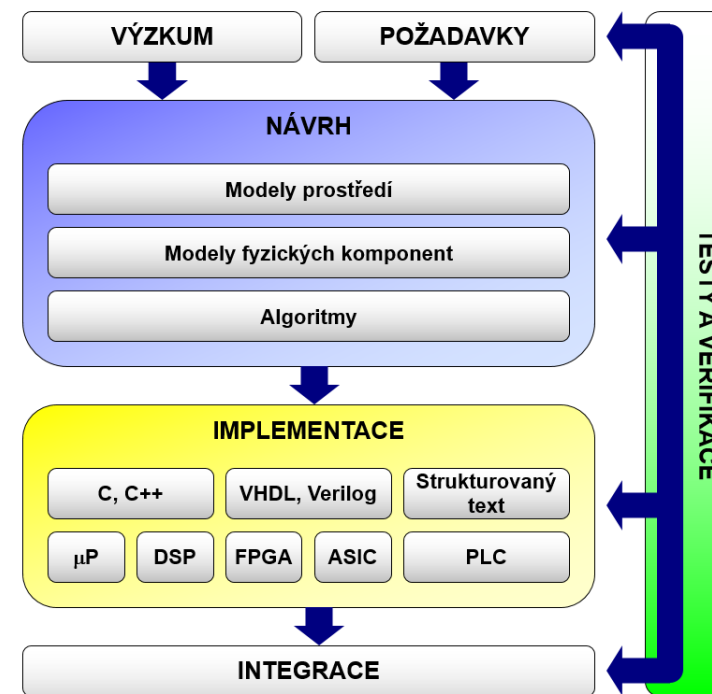
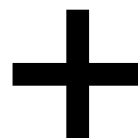
Show more

Sequence Networks

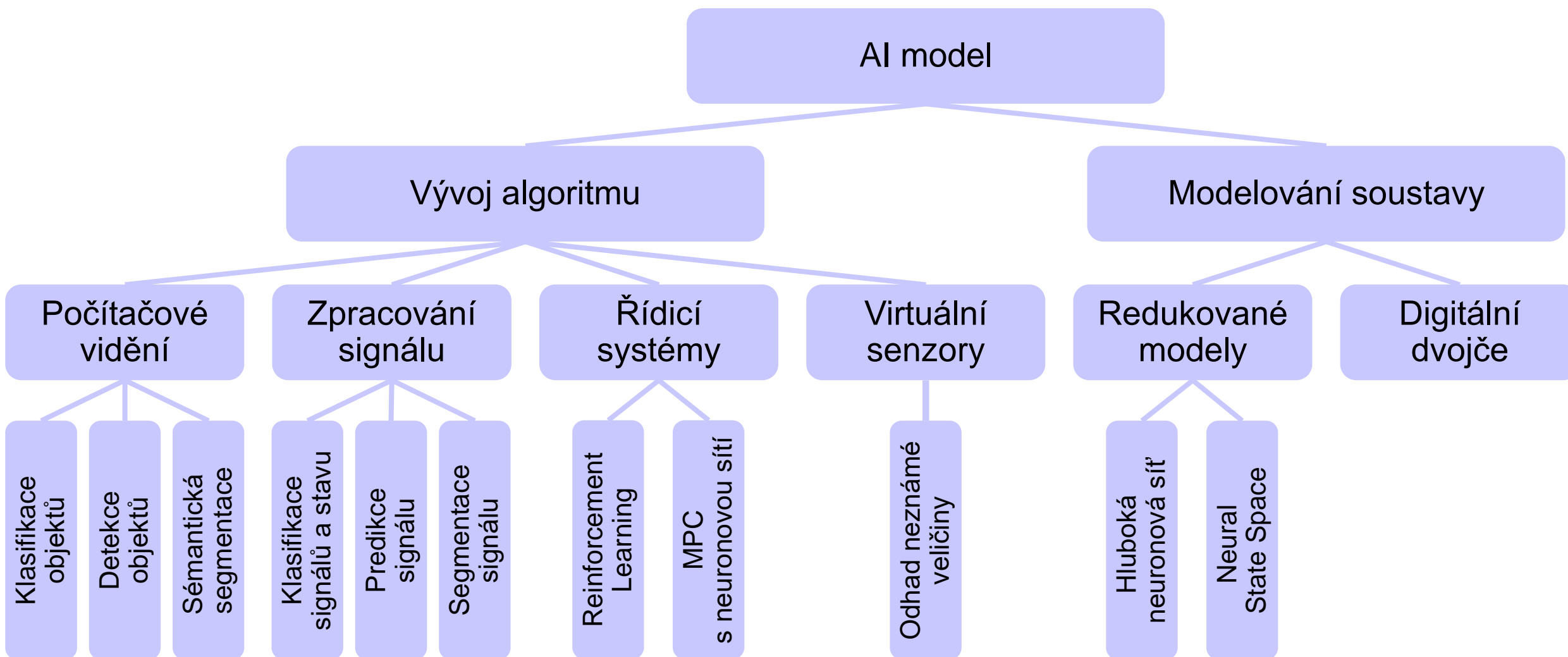


Využit předdefinované sítě a před-učené sítě

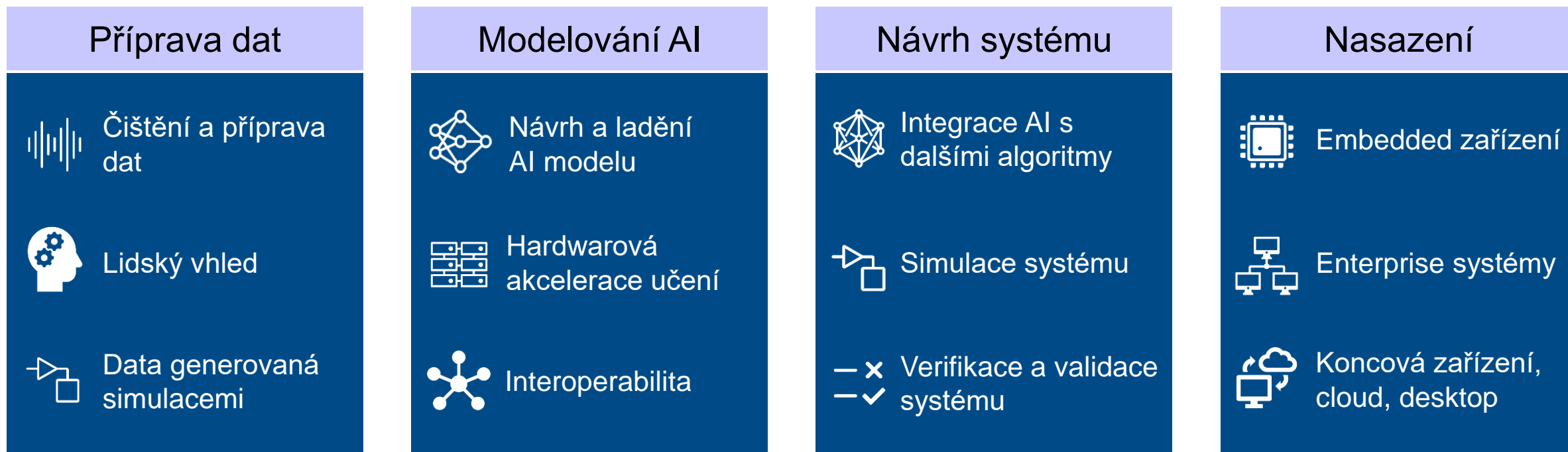
AI a Model-Based Design



Modelování AI – Typ algoritmu



Postup návrhu systému využívajícího AI

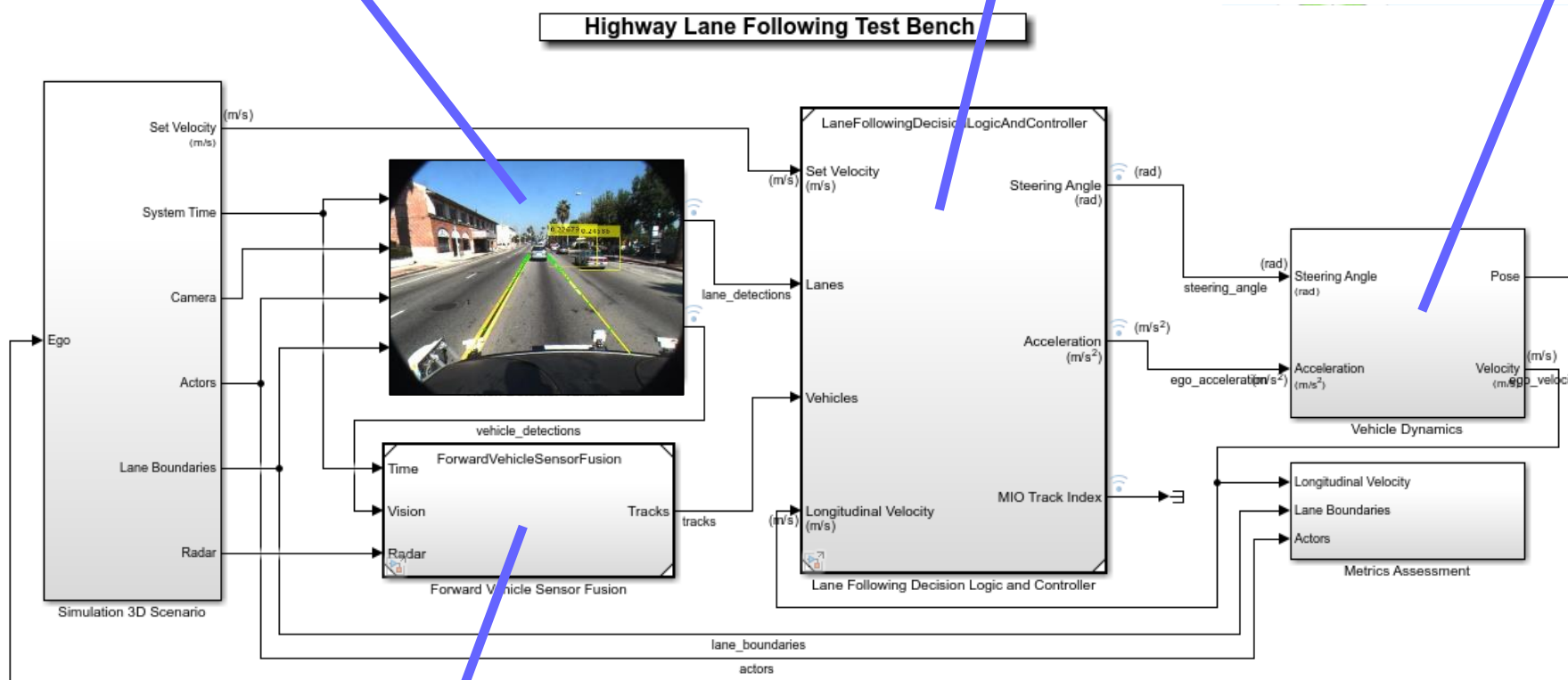


Typický příklad: Asistenční systém automobilu

AI algoritmus
detekce jízdnic a vozidel

Řídicí systém

Model dynamiky vozidla
pro testování chování systému



Senzorická fúze

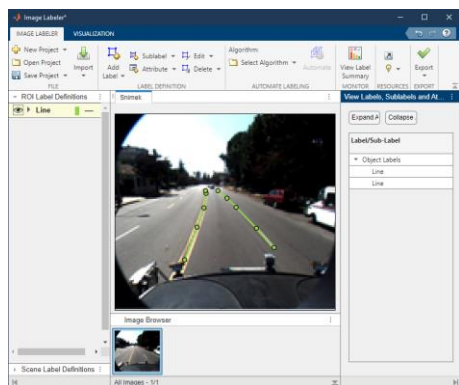
Typický příklad: Asistenční systém automobilu

Příprava dat

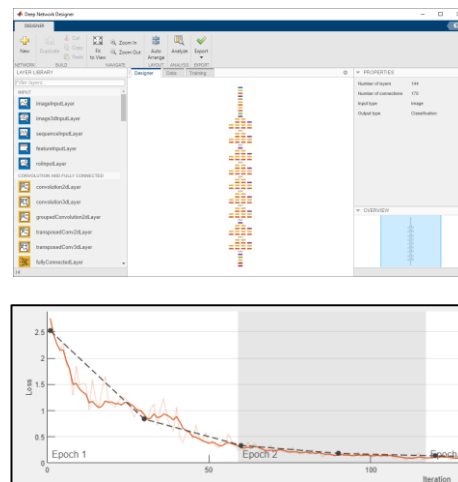
Modelování AI

Návrh systému

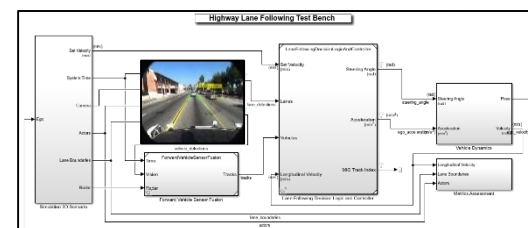
Nasazení



- 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



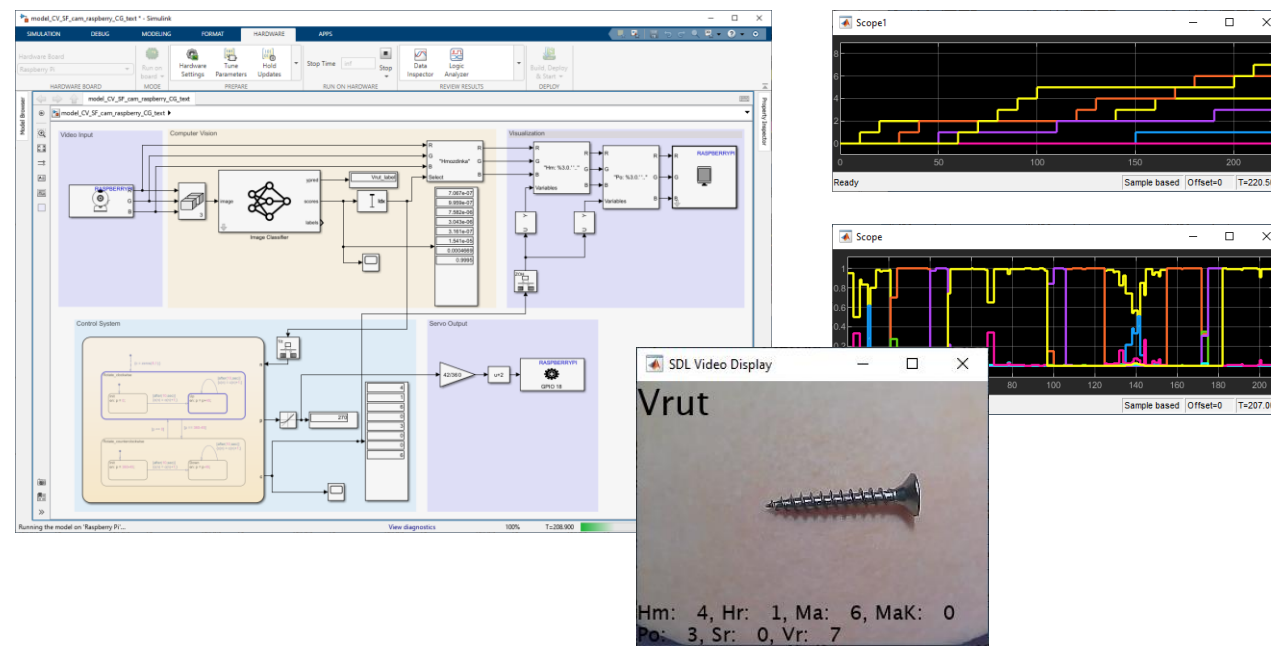
- propojení AI modelů s ostatními algoritmy
- simulace a testování modelu celého software



- generování kódu pro cílovou embedded platformu

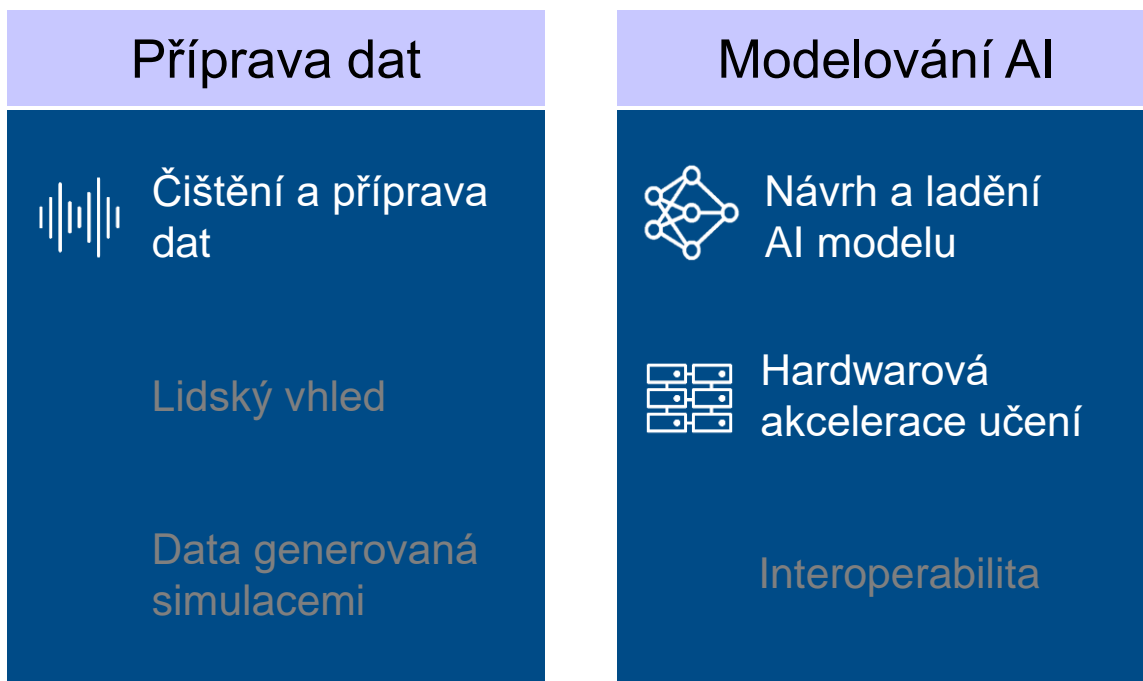
Ukázka: Systém pro klasifikaci a počítání objektů

- Klasifikační algoritmus založený na AI
 - blok pro inferenci naučeného modelu z knihovny Deep Learning Toolbox
- Řídicí systém
 - stavový model v jazyce Stateflow
 - řízení otáčení stolu
 - počítání stejných objektů
- Spuštění a nasazení algoritmu
 - v testovacím režimu
 - v režimu samostatné aplikace
- Hardware
 - Raspberry Pi 4, webkamera, servomotor

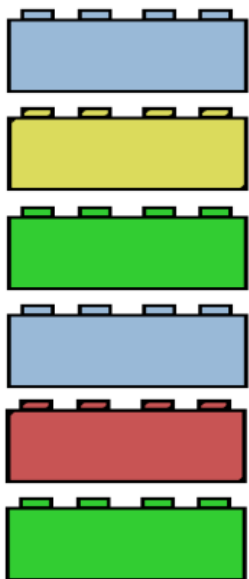


Ukázka: Systém pro klasifikaci a počítání objektů

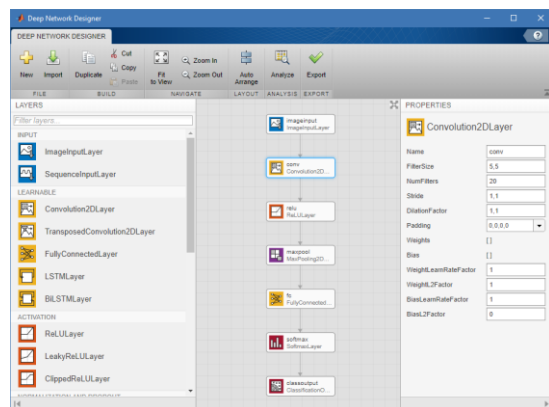
- 1. vytvoření AI modelu pro klasifikaci objektů z obrazových dat



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

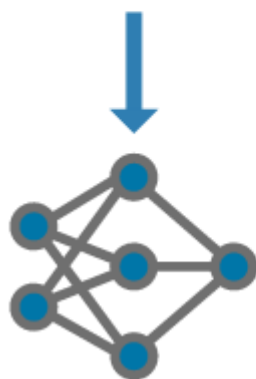
uživatelské smyčky pro učení sítí
automatická diferenciace
sdílené váhy
uživatelské ztrátové funkce
...

GAN, CGAN, siamské sítě, ...

Transfer learning

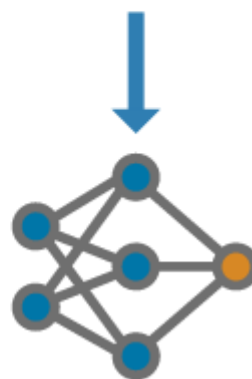
- Využití předdefinovaných a před-učených sítí

datová sada 1



model 1

datová sada 2

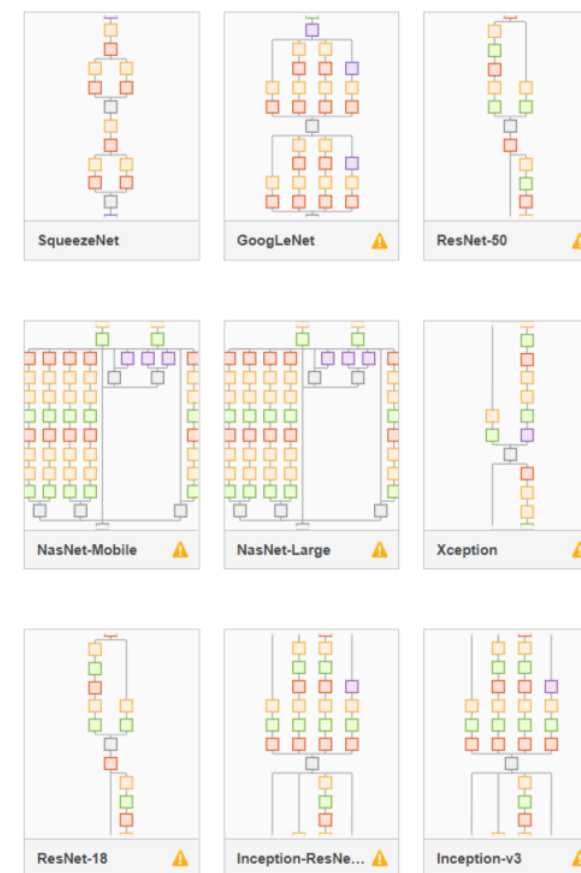


model 2

„znalost“ síť

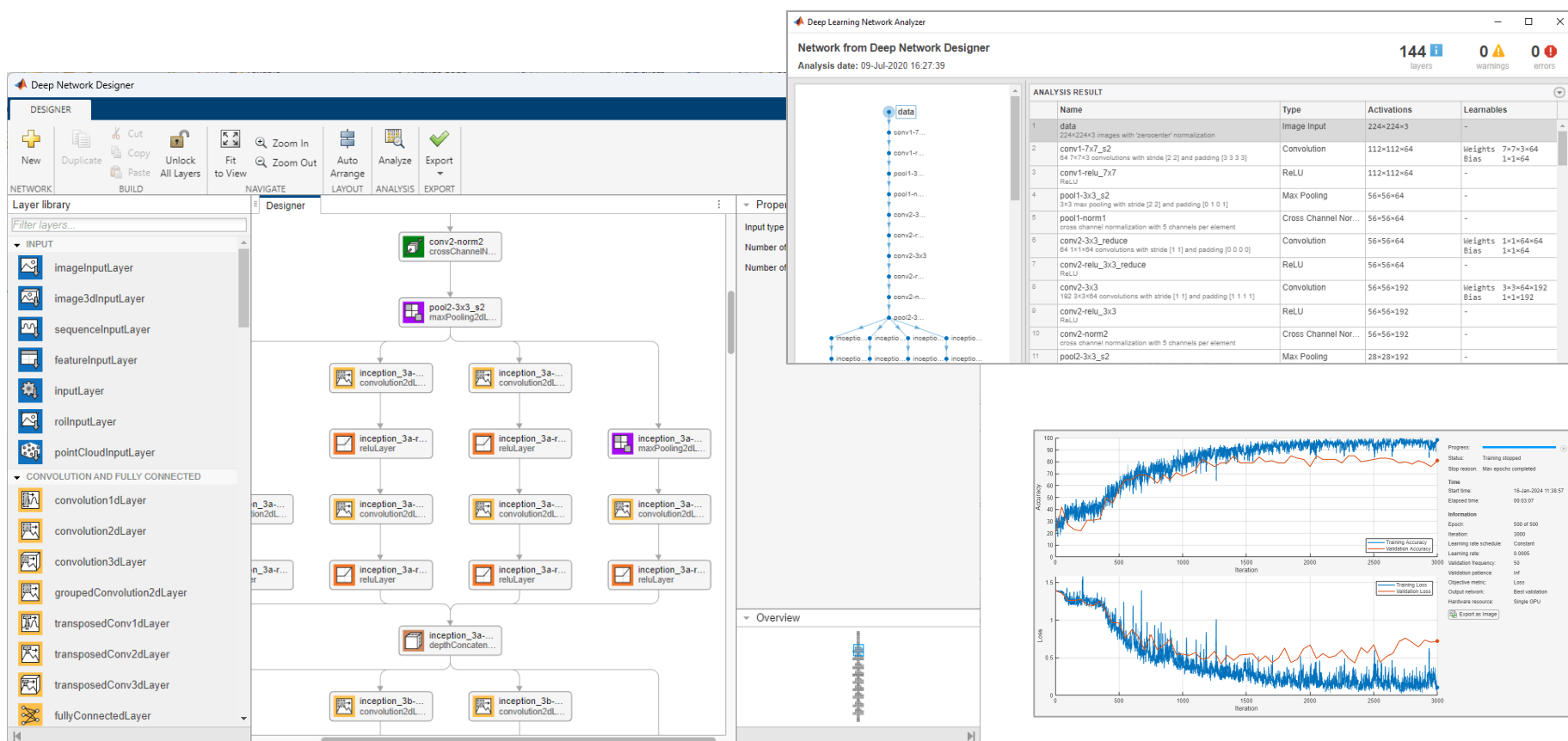


Pretrained Networks



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 Design Canvas: Shows a hierarchical flowchart of a neural network. The input layer is followed by a convolutional layer with 224x224x3 filters, a max pooling layer (3x3), and an Inception v3 architecture consisting of multiple parallel branches of convolutional and pooling layers.

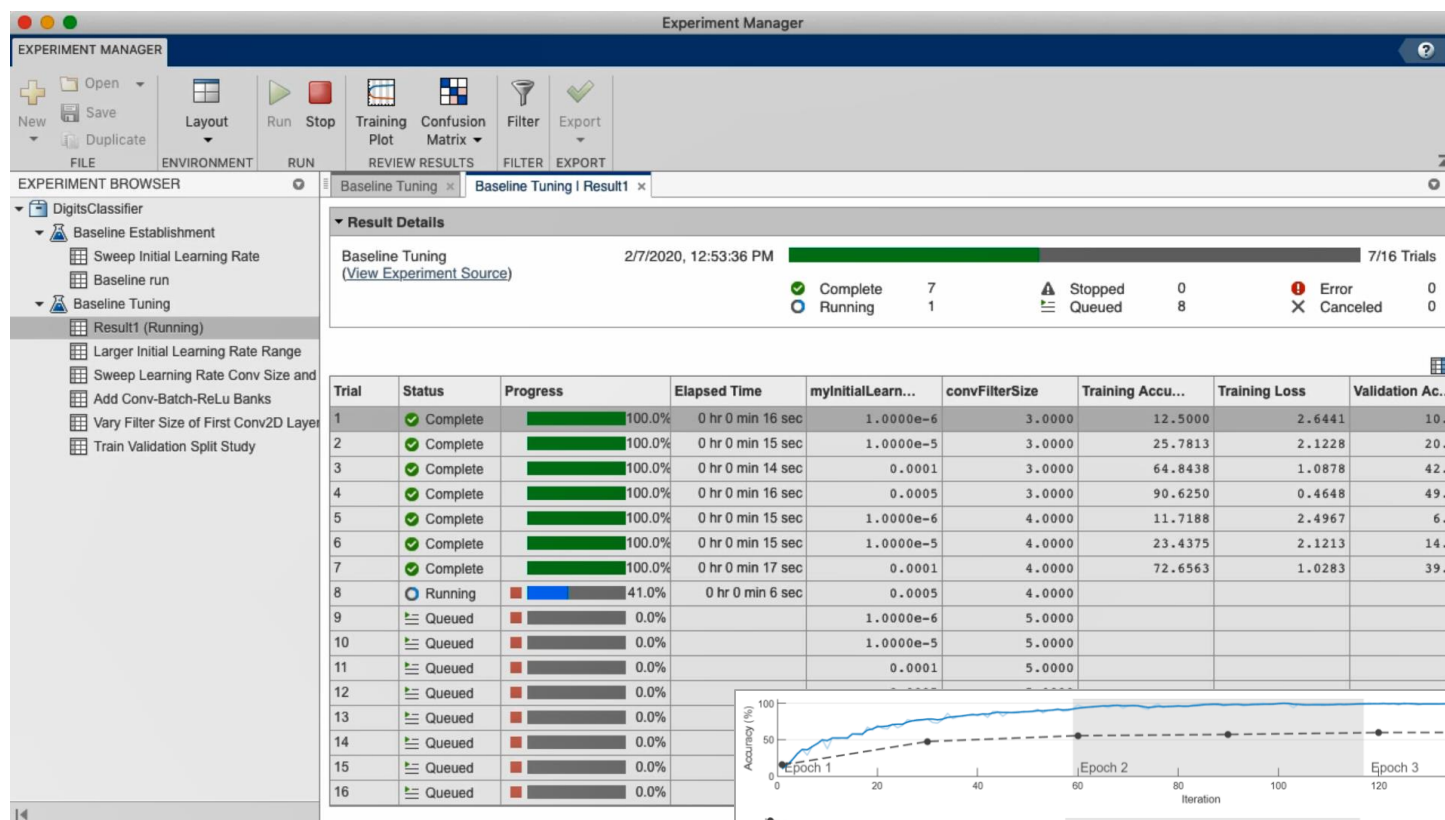
Analysis Window: Titled "Deep Learning Network Analyzer", it provides a summary of the network's performance. The analysis date is 09-Jul-2020 16:27:39. The network consists of 144 layers, with 0 warnings and 0 errors.

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

Training Progress Graphs: Two line graphs are shown. The top graph plots Accuracy (0-100) against Iteration (0-3000). Training accuracy (blue) and validation accuracy (orange) both rise and stabilize around 80% after approximately 1000 iterations. The bottom graph plots Loss (0-1.5) against Iteration (0-3000). Training loss (blue) and validation loss (orange) both decrease and stabilize around 0.5 after approximately 1000 iterations. The training status is "Training stopped" with "Max epochs completed" as the reason.

Experiment Manager

- Nalezení optimální sítě pomocí experimentů



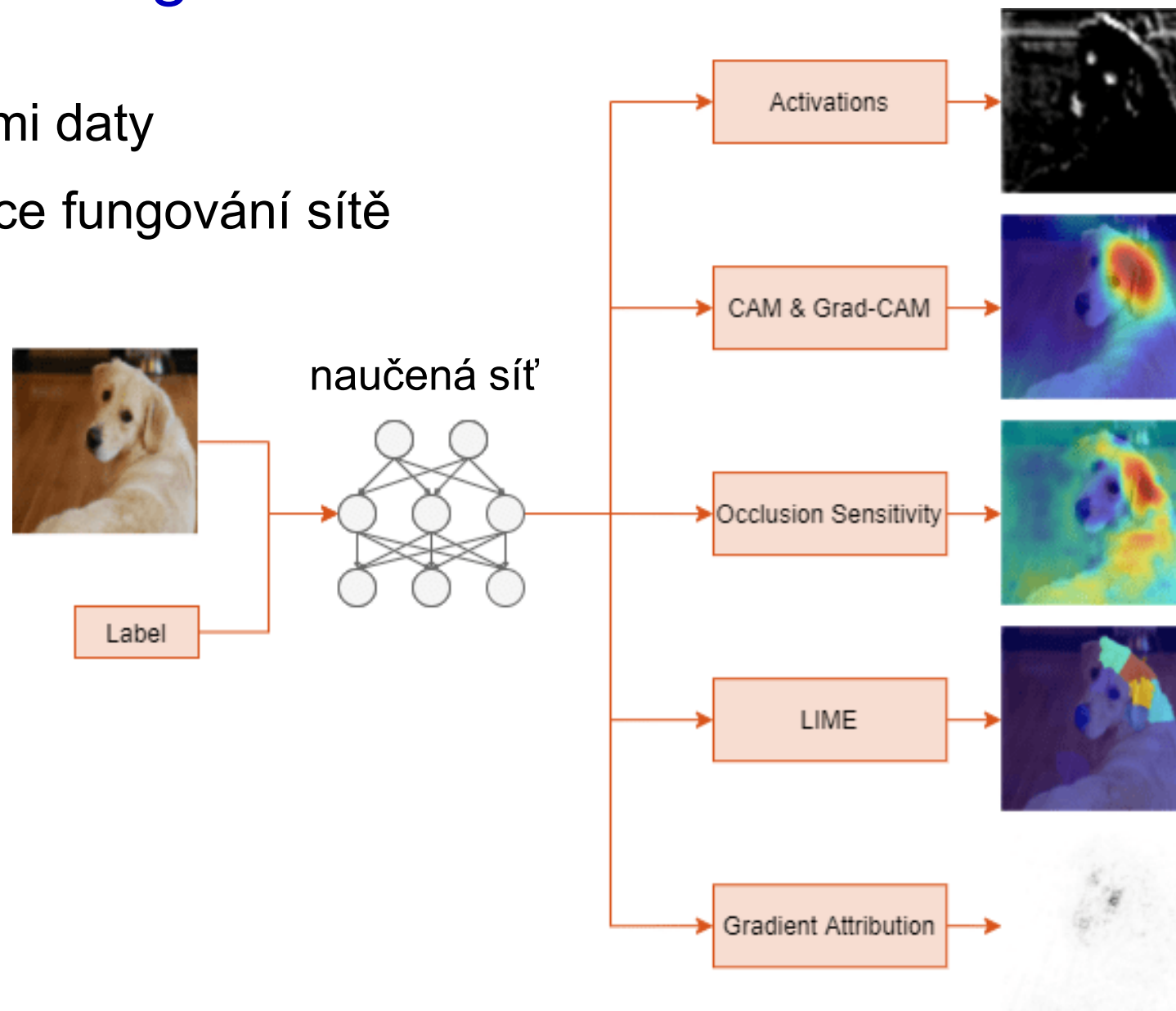
The screenshot displays the Experiment Manager interface. On the left is the 'EXPERIMENT BROWSER' showing a tree structure for 'DigitsClassifier' with sub-items like 'Baseline Establishment', 'Baseline Tuning', and 'Result1 (Running)'. The main area shows 'Result Details' for 'Baseline Tuning' on 2/7/2020 at 12:53:36 PM, indicating 7/16 trials. 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 three epochs. The bottom graph shows Loss decreasing from ~2.0 to ~0.5 over the same period.

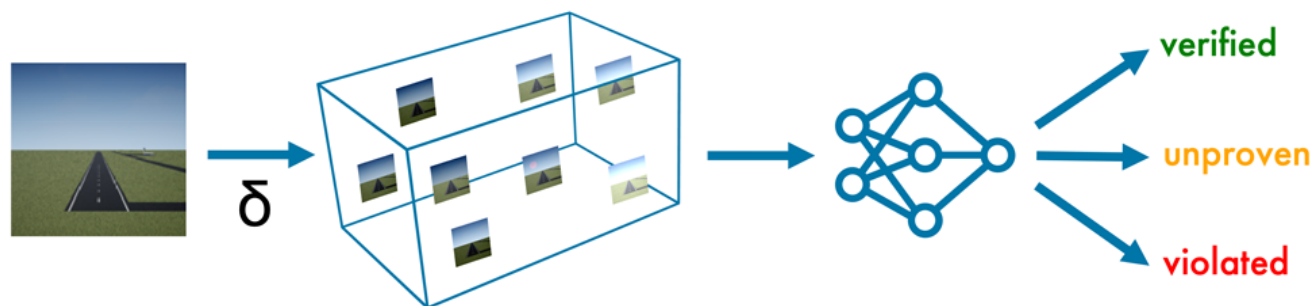
Ověření správného fungování sítě

- 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

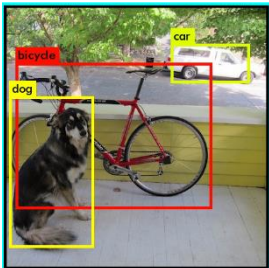
Jak zahrnout deep learning do Simulinku

- Připravené bloky pro deep learning

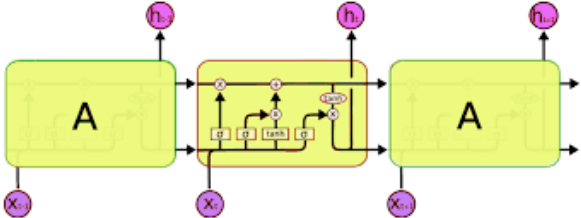


leopard

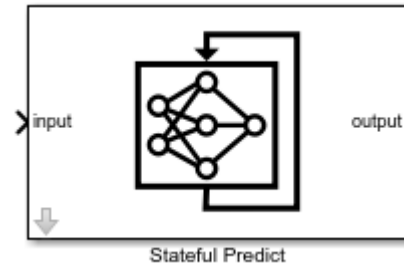
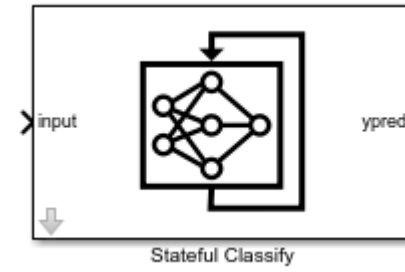
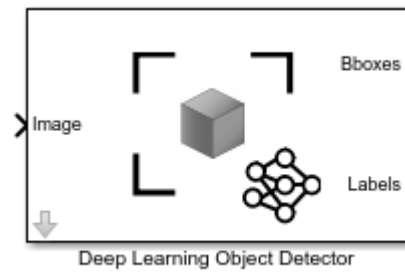
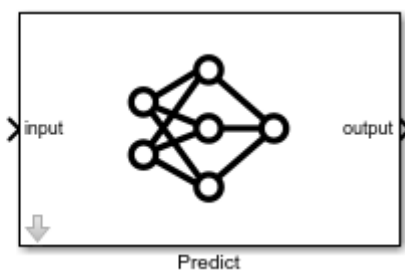
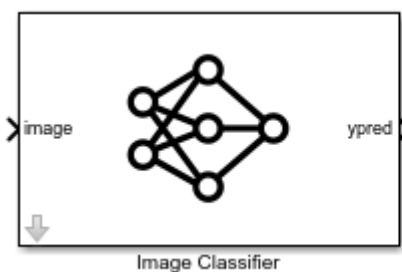
**Klasifikace obrazu,
sémantická segmentace**



Detektory objektů

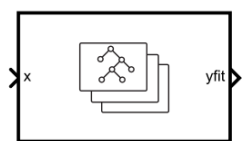


**Sekvenční síť pro zvuková,
textová a signálová data**



- Blok MATLAB Function

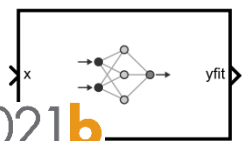
Bloky pro AI v Simulinku stále přibývají



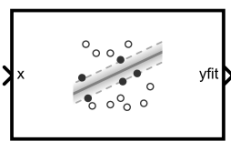
RegressionEnsemble Predict



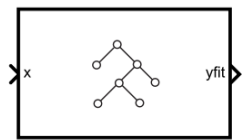
RegressionGP Predict



RegressionNeuralNetwork Predict



RegressionSVM Predict



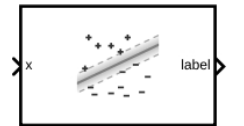
RegressionTree Predict



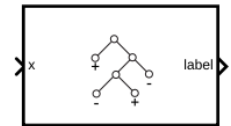
ClassificationEnsemble Predict



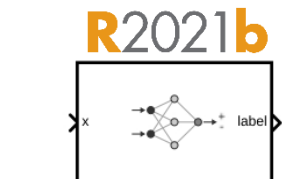
ClassificationKNN Predict



ClassificationSVM Predict



ClassificationTree Predict



ClassificationNeuralNetwork Predict

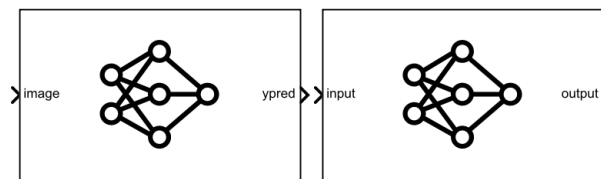
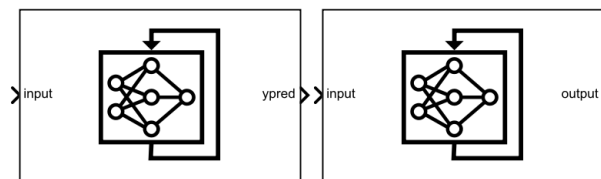


Image Classifier

Predict



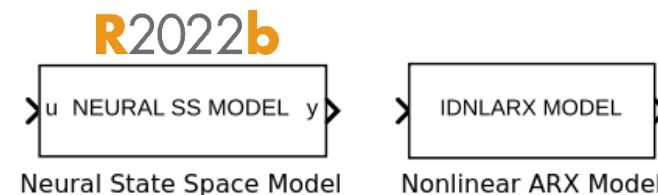
Stateful Classify

Stateful Predict

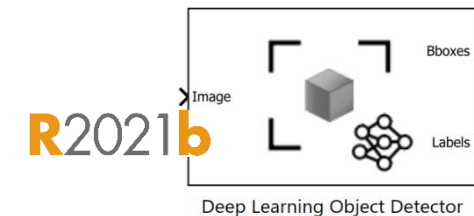
Deep Learning Toolbox



Audio Toolbox



System Identification Toolbox

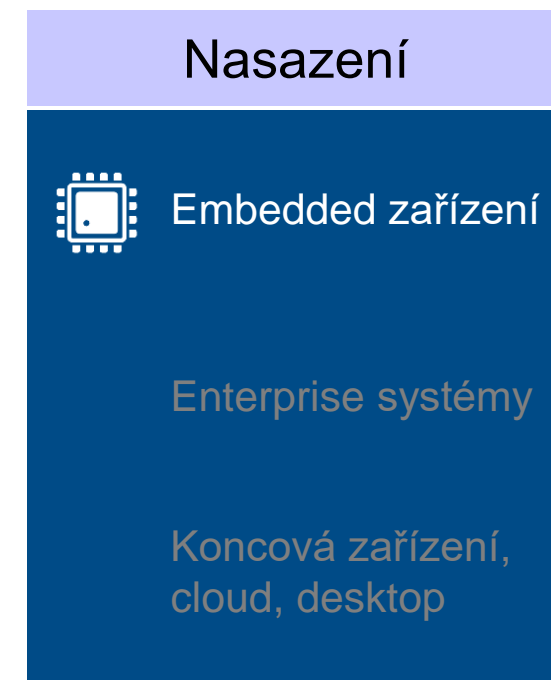


Computer Vision Toolbox

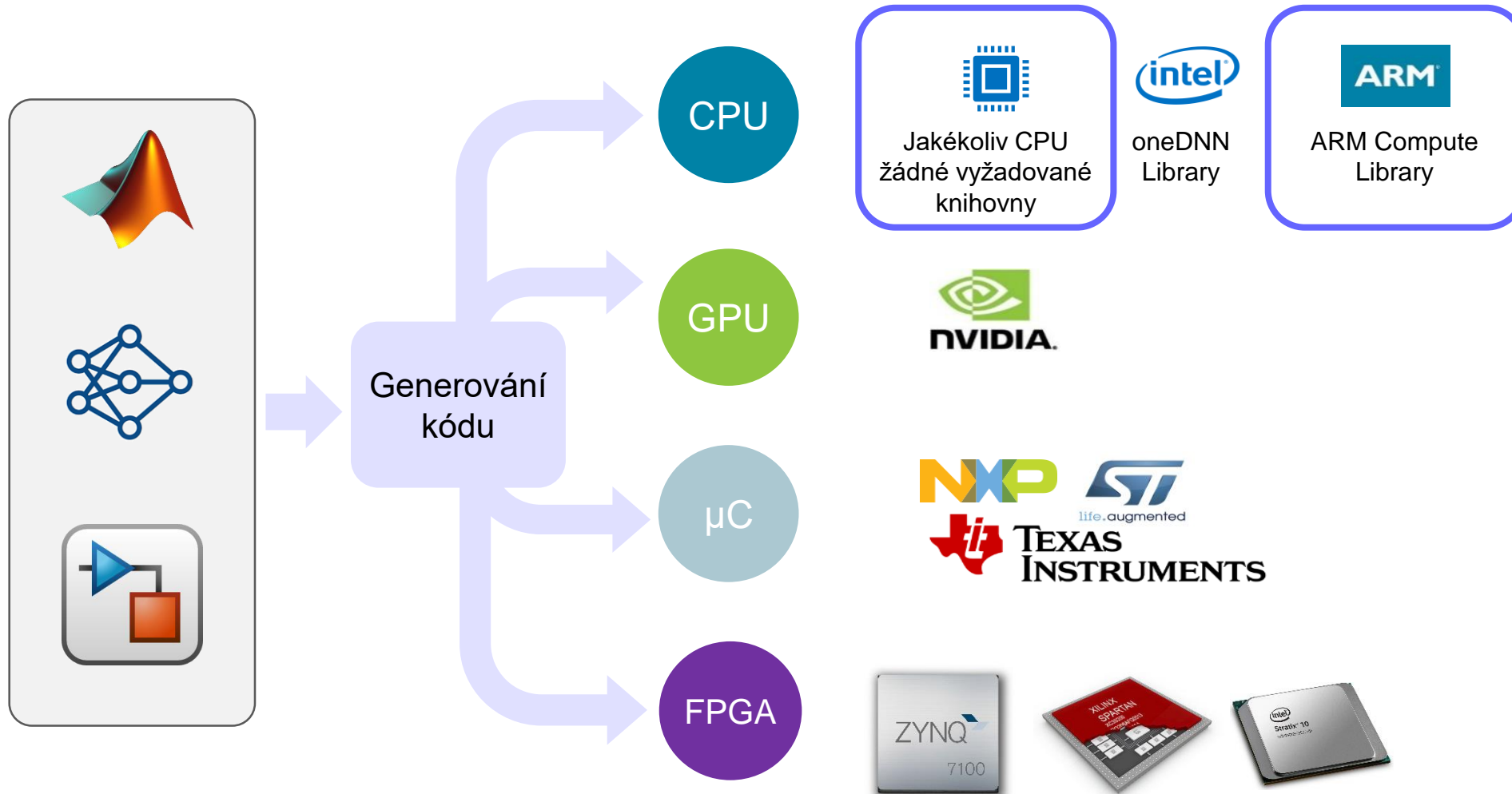
Statistics and Machine Learning Toolbox

Ukázka: Systém pro klasifikaci a počítání objektů

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



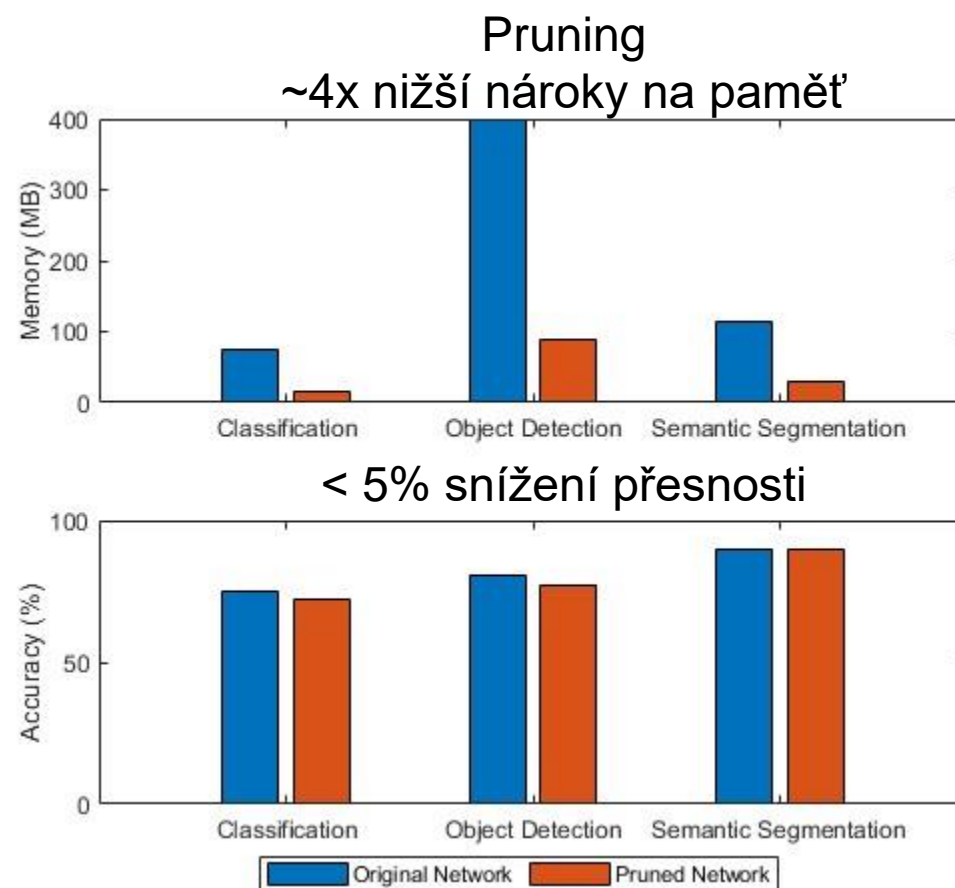
Nasazení na cílovou platformu



Redukce modelu

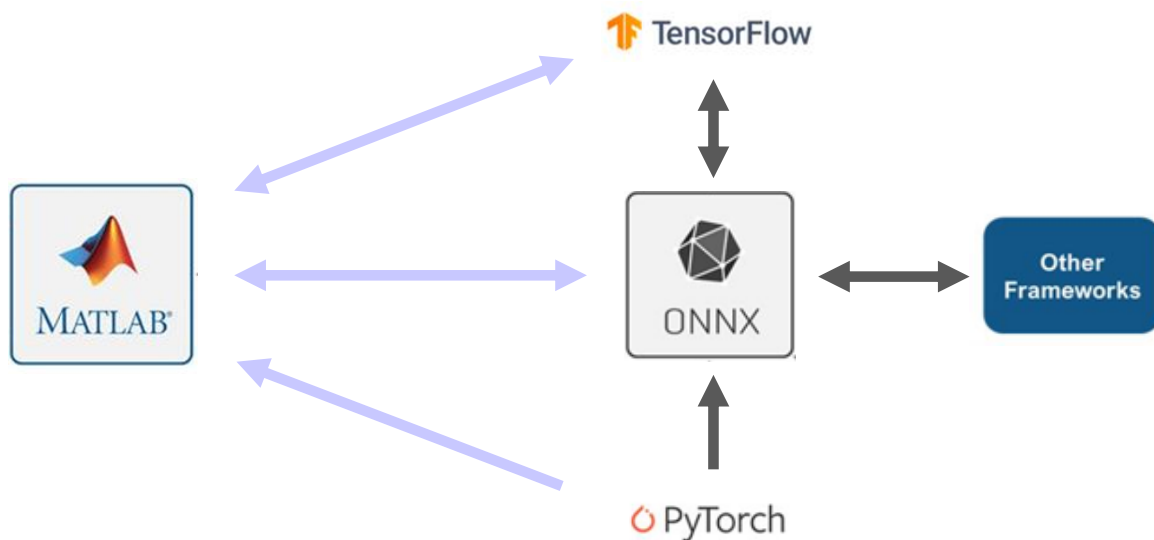
- Snížení paměťových a výpočetních nároků nasazeného modelu
- **Quantizace**
 - převod z floating point do fixed point aritmetiky
- **Pruning**
 - odstranění nedůležitých částí sítě

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



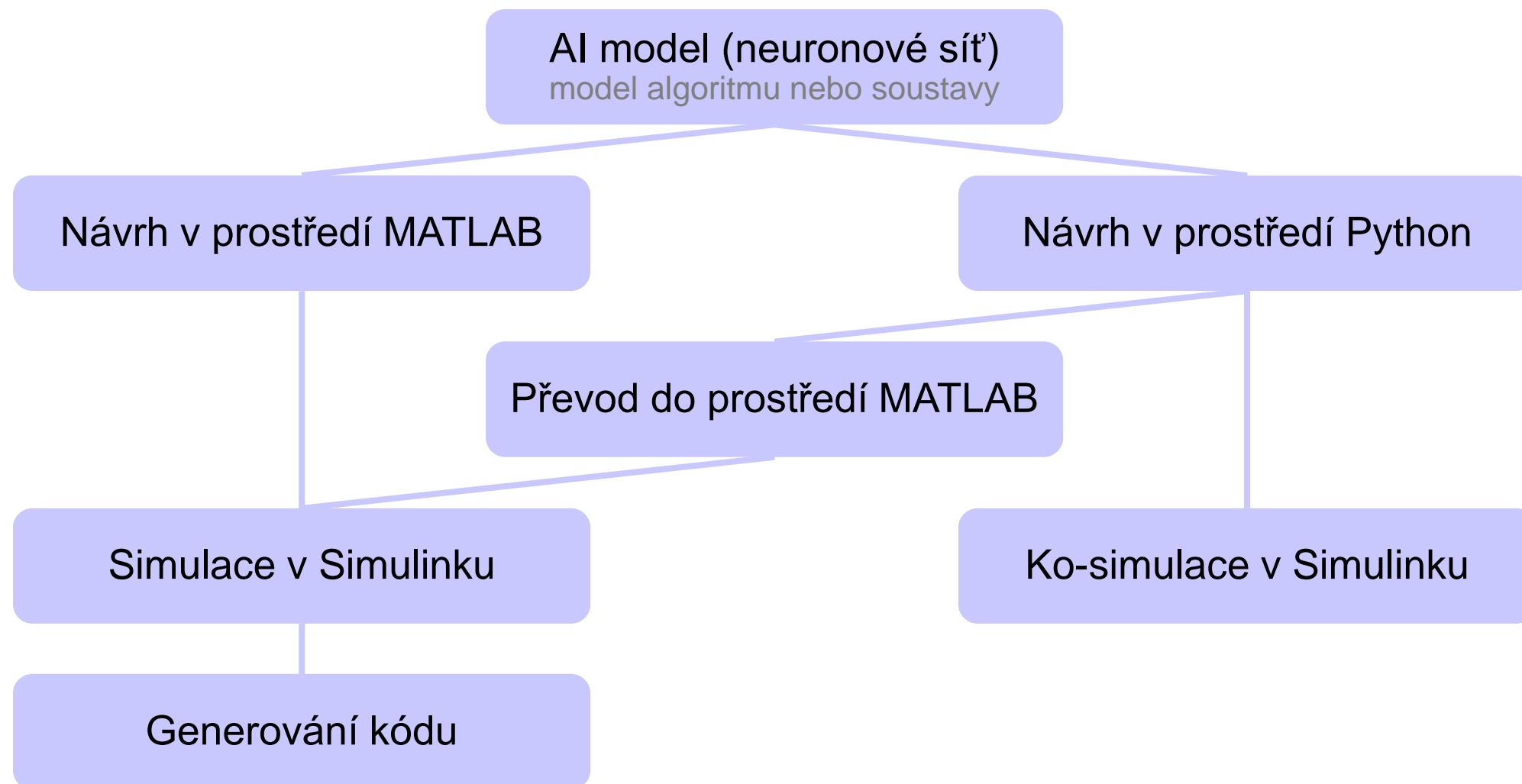
MATLAB a spolupráce s dalšími nástroji

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



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

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

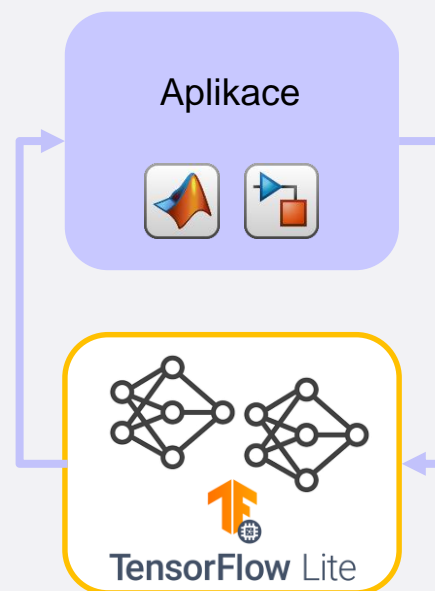
R2022b

Support for TFLite 2.8.0

R2023a

Simulace

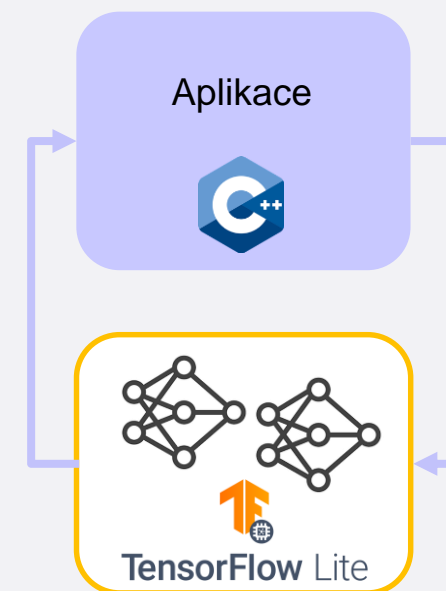
Aplikace spuštěná v
MATLAB/Simulink



DL síť spuštěná v TFLite

Generování kódu

Aplikace ve formě
generovaného C++ kódu



DL síť spuštěná v TFLite

Otázky