

Általános információk, a diplomaterv szerkezete

A diplomaterv szerkezete a BME Villamosmérnöki és Informatikai Karán:

1. Diplomaterv feladatkiírás
2. Címoldal
3. Tartalomjegyzék
4. A diplomatervező nyilatkozata az önálló munkáról és az elektronikus adatok kezeléséről
5. Tartalmi összefoglaló magyarul és angolul
6. Bevezetés: a feladat értelmezése, a tervezés célja, a feladat indokoltsága, a diplomaterv felépítésének rövid összefoglalása
7. A feladatkiírás pontosítása és részletes elemzése
8. Előzmények (irodalomkutatás, hasonló alkotások), az ezekből levonható következtetések
9. A tervezés részletes leírása, a döntési lehetőségek értékelése és a választott megoldások indoklása
10. A megtervezett műszaki alkotás értékelése, kritikai elemzése, továbbfejlesztési lehetőségek
11. Esetleges köszönhetők
12. Részletes és pontos irodalomjegyzék
13. Függelék(ek)

Felhasználható a következő oldaltól kezdődő L^AT_EX-diplomatervsablon dokumentum tartalma.

A diplomaterv szabványos méretű A4-es lapokra kerüljön. Az oldalak tükörmagjával készüljenek (mindenhol 2,5 cm, baloldalon 1 cm-es kötéssel). Az alapértelmezett betűkészlet a 12 pontos Times New Roman, másfeles sorközzel, de ettől kismértékben el lehet tértíni, ill. más betűtípus használata is megengedett.

Minden oldalon – az első négy szerkezeti elem kivételével – szerepelnie kell az oldalszámnak.

A fejezeteket decimális beosztással kell ellátni. Az ábrákat a megfelelő helyre be kell illeszteni, fejezetenként decimális számmal és kifejező címmel kell ellátni. A fejezeteket decimális aláosztással számozzuk, maximálisan 3 aláosztás mélységen (pl. 2.3.4.1.). Az ábrákat, táblázatokat és képleteket célszerű fejezetenként külön számozni (pl. 2.4. ábra, 4.2. táblázat vagy képletnél (3.2)). A fejezetcímeket igazitsuk balra, a normál szövegnél viszont használunk sorkiegynítést. Az ábrákat, táblázatokat és a hozzájuk tartozó címet igazitsuk középre. A cím a jelölt rész alatt helyezkedjen el.

A képeket lehetőleg rajzoló programmal készítsék el, az egyenleteket egyenlet-szerkesztő segítségével írják le (A L^AT_EX ehhez kézenfekvő megoldásokat nyújt).

Az irodalomjegyzék szövegközi hivatkozása történhet sorszámozva (ez a preferált megoldás) vagy a Harvard-rendszerben (a szerző és az évszám megadásával). A teljes lista névsor szerinti sorrendben a szöveg végén szerepeljen (sorszámozott irodalmi hivatkozások esetén hivatkozási sorrendben). A szakirodalmi források címeit azonban mindenkor nyelven kell megadni, esetleg zárójelben a fordítással. A listában szereplő valamennyi publikációra hivatkozni kell a szövegben (a L^AT_EX-sablon a Bib^TEX segítségével mindenkor automatikusan kezeli). minden publikáció a szerzők után a következő adatok szerepelnek: folyóirat cikkeknél a pontos cím, a folyóirat címe, évfolyam, szám, oldalszám tól-ig. A folyóiratok címét csak akkor rövidítsük, ha azok nagyon közismertek vagy nagyon hosszúak. Internetes hivatkozások megadásakor fontos, hogy az elérési út előtt megadjuk az oldal tulajdonosát és tartalmát (mivel a link egy idő után akár elérhetetlenne is válhat), valamint az elérés időpontját.

Fontos:

- A szakdolgozatkészítő / diplomatervező nyilatkozata (a jelen sablonban szereplő szövegtartalommal) kötelező előírás, Karunkon ennek hiányában a szakdolgozat/diplomaterv nem bírálható és nem védehető!
- Mind a dolgozat, mind a melléklet maximálisan 15 MB méretű lehet!

Jó munkát, sikeres szakdolgozatkészítést, ill. diplomatervezést kívánunk!

FELADATKIÍRÁS

A feladatkiírást a tanszéki adminisztrációban lehet átvenni, és a leadott munkába eredeti, tanszéki pecséttel ellátott és a tanszékvezető által aláírt lapot kell belefűzni (ezen oldal helyett, ez az oldal csak útmutatás). Az elektronikusan feltöltött dolgozatban már nem kell beleszerkeszteni ezt a feladatkiírást.



Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics
Department of Control Engineering and Information Technology

Comparison of convolution and transformer-based image processing neural networks

BACHELOR'S THESIS

Author
István Péter

Advisor
dr. Bálint Kiss
Ádám Gyula Nemes

December 3, 2022

Contents

Kivonat	i
Abstract	ii
1 Introduction	1
1.1 The field of Computer Vision	1
1.2 Object Detection	1
1.3 My Goal	1
2 Overview of the Literature	2
2.1 YOLO: You Only Look Once	2
2.1.1 Implementation details	2
2.2 DETR: The Detection Transformer	2
2.2.1 The Transformer in Natural Language Processing	2
2.2.1.1 Multi	3
2.2.1.2 What is the difference between a neural layer and the attention?	3
2.2.2 DETR	3
2.2.3 Explainability	3
2.3 Comparison	3
3 Practical Applications: Training on a Specific Task	4
3.1 Datasets	4
3.1.1 Preparing the filtered COCO dataset	4
4 Higher Order Applications: Multiple Object Tracking	6
4.1 The Problem	6
4.2 Metrics	7
4.3 Solutions	7
4.4 My measurement	7
4.4.1 Data	7

4.4.2	Benchmark	7
5	A L^AT_EX-sablon használata	9
5.1	Címkék és hivatkozások	9
5.2	Ábrák és táblázatok	9
5.3	Felsorolások és listák	11
5.4	Képletek	12
5.5	Irodalmi hivatkozások	13
5.6	A dolgozat szerkezete és a forrásfájlok	16
5.7	Alapadatok megadása	18
5.8	Új fejezet írása	18
5.9	Definíciók, tételek, példák	18
	Bibliography	19

HALLGATÓI NYILATKOZAT

Alulírott *Péter István*, szigorló hallgató kijelentem, hogy ezt a szakdolgozatot meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy autentikált felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Budapest, 2022. december 3.

Péter István
hallgató

Kivonat

Jelen dokumentum egy diplomaterv sablon, amely formai keretet ad a BME Villamosmérnöki és Informatikai Karán végző hallgatók által elkészítendő szakdolgozatnak és diplomaternevnek. A sablon használata opcionális. Ez a sablon L^AT_EX alapú, a *TeXLive* T_EX-implementációval és a PDF-L^AT_EX fordítóval működőképes.

Abstract

This document is a L^AT_EX-based skeleton for BSc/MSc theses of students at the Electrical Engineering and Informatics Faculty, Budapest University of Technology and Economics. The usage of this skeleton is optional. It has been tested with the *TeXLive* T_EX implementation, and it requires the PDF-L^AT_EX compiler.

Chapter 1

Introduction

A bevezető tartalmazza a diplomaterv-kiírás elemzését, történelmi előzményeit, a feladat indokoltságát (a motiváció leírását), az eddigi megoldásokat, és ennek tükrében a hallgató megoldásának összefoglalását.

A bevezető szokás szerint a diplomaterv felépítésével záródik, azaz annak rövid leírásával, hogy melyik fejezet mivel foglalkozik.

1.1 The field of Computer Vision

Computer Vision is a branch of Artificial Intelligence aimed at deducting higher-order information from visual input like images, videos, or more specialised sensor data like LiDAR point clouds etc. Some individual tasks in Computer Vision are image classification, object detection, segmentation, pose estimation of specific entities etc.

1.2 Object Detection

Blah Blah

1.3 My Goal

My goal in this thesis is to give an overview of the differences between already established Fully Convolutional Neural Networks (FCN) compared to upcoming Transformer-based architectures in the task of Object Detection. In the former, I restrict myself to single-stage detectors, namely the YOLO architecture.

Chapter 2

Overview of the Literature

In this chapter I am going to review the theoretical background for the two competing paradigms I cover: the fully convolutional, one-stage detector, whose most prominent variant is the You Only Look Once (YOLO) architecture, and the Transformer-based Detectection Transformer (DETR). For the former, I will explain in some detail chosing it over its competitors of the same kind, for example the Single Shot Detector (SSD).

In the case of the Tranformer-based category, I chose, for the sake of simplicity, the DETR architecture over its later successors, like DINO or Deformable DETR. The changes introduced in **its paper (insert citation)** are important enough to be discussed on their own, but I will mention the improvements achieved by the successors whenever the state-of-the-art is concerned.

Likewise, I have chosen the YOLOv5 for in-depth comparison as the DETR's counterpart, mainly because it is a contemporary of the latter (both being introduced in 2020), but mentioning the latest improvements introduced by YOLOv7 as well.

2.1 YOLO: You Only Look Once

2.1.1 Implementation details

2.2 DETR: The Detection Transformer

2.2.1 The Transformer in Natural Language Processing

The Transformer architecture has been introduced in the *Attention is All You Need* [4] paper in 2017, originally intended for Natural Language Processing (NLP) tasks, more specifically sequence transduction problems, like translation.

At the time, the attention mechanism and some variants of the encoder-decoder architecture was already widely used in the state-of-the-art, along with convolutional layers, Long Short Term Memory (LSTM) cells or Gated Recurrent Units (GRU). The Transformer was a successful attempt at replacing the latter three with trainable versions of the attention mechanisms called *Multi-Head Attention*.

In the Transformer model, the bulk of the learning happens at the weights of the linear transformations that estabilish the **heads** of the Attention layers, as the Attention layer itself does only mathematical operations on its input.

The article mentions that attention mechanisms and encoder-decoder based architectures have already been used at the time in the state-of-the art models. The novelty of the Transformer was getting rid of the convolutional, or traditionally recurrent components, and relying almost solely on the attention mechanism, namely a slightly modified version of it: the *multi-head self-attention*.

2.2.1.1 Multi

2.2.1.2 What is the difference between a neural layer and the attention?

2.2.2 DETR

The main advantage of the Detection Transformer is its capacity for every region to attend to every other region. In the fully convolutional case, this is done with hierarchical convolutions that together define large *receptive fields*.

2.2.3 Explainability

2.3 Comparison

Chapter 3

Practical Applications: Training on a Specific Task

3.1 Datasets

The Microsoft COCO: Common Objects in Context is an immensely popular dataset for numerous computer vision tasks (classification, object detection, instance segmentation, collectively called object recognition). It was introduced in 2014 in the paper *Microsoft COCO: Common Objects in Context* [2], aiming to improve upon already existent visual datasets like ImageNet and PASCAL, and striving to be a benchmark of scene understanding. It contains labeled data for 91 object classes, captured in their natural habitat (hence *context*). It was later updated in 2017, a notable change being the introduction of *stuff* labels (among the already existent *thing* labels), for objects with no clear boundaries, like sky and grass. These were introduced as panoptic segmentation labels.

The official site of the dataset¹ mentions that the creators of COCO have partnered with the developers of open source tool FiftyOne in providing a software to facilitate downloading, visualizing, and evaluation on COCO, so I will be using it for initial data exploration. Another tool endorsed by the creators is the COCO API.

I conducted initial data exploration on the COCO dataset with the FiftyOne tool (launching an application session from a Python shell as described in their tutorials), as seen in figure 3.1. The success of the COCO dataset lies in its richness and difficulty: some training images, as seen in figure 3.2, are caught in so called *non-iconic views*, where visual features can be ambiguous and they should be interpreted in context to recognise the object.

3.1.1 Preparing the filtered COCO dataset

¹<https://cocodataset.org>

Figure 3.1: Exploring the train split of the MS COCO 2017 dataset in the FiftyOne tool. There are 43867 instances of labeled cars and 9973 instances of labeled trucks, with 14714 images containing either of them.

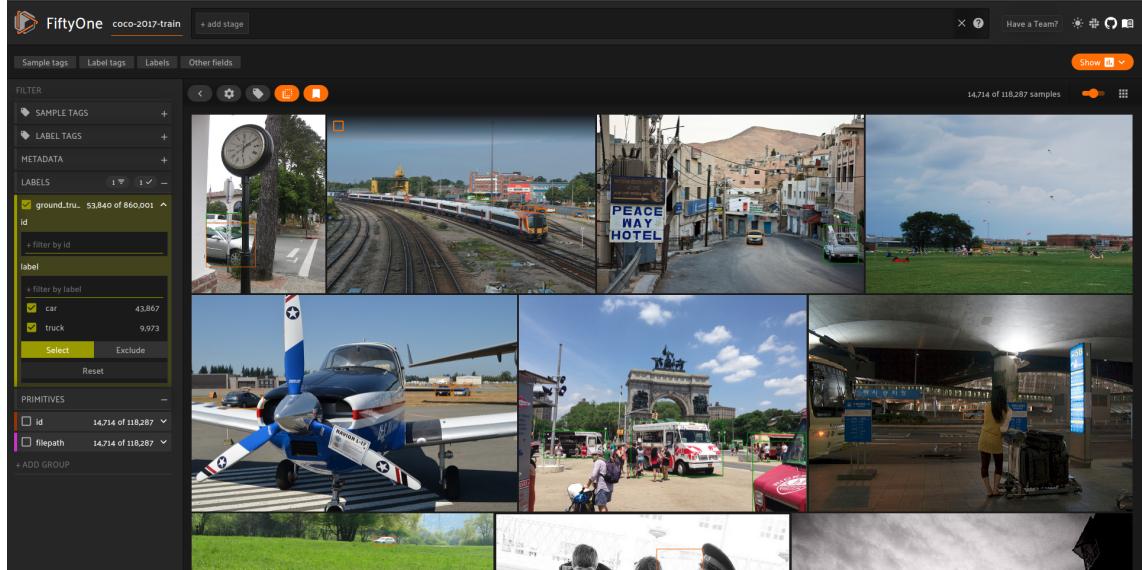


Figure 3.2: Some difficult examples: objects in overexposed regions of the image, crowd labels consisting of very small (e.g aerial) images of cars (orange) and trucks(green), detection based on a skewed reflection of the object, small parts of it.



Chapter 4

Higher Order Applications: Multiple Object Tracking

As the final aim of my thesis, I am going to inspect the performance of the chosen object detectors on a higher-order problem, namely multiple object tracking. First I am going to review the problem itself, and the common metrics used to measure its performance.

After this, I will elaborate on the object tracking method I chose as the basis, showing how it incorporates the aforementioned detectors, and the way the performance of the latter is expected to influence tracking performance.

4.1 The Problem

Multiple object tracking consists of determining the *trajectory* of given kinds of objects in a video stream. In the current context, the trajectory of a single, unique instance is a series of bounding boxes, one for every frame in which the object is visible. The boxes are expected to fit the object as tightly as possible, and the trajectories have usually some kind of identity associated with them.

The problem builds on detection, but the difficulty also comes in assigning the current detected objects a correct identity, if it has been seen previously, or a new one otherwise. This action is generally called *association*. There are usually two, not necessarily exclusive approaches: association between detections in subsequent frames (the identified objects in the previous frames are expected to be close to their current location, thus associating by proximity), or overarching association by some appearance features (association by feature similarity).

Proximity-based association fails in scenarios where multiple objects of the same kind are very close to each other and dynamically moving, while visible features can vary because of the lighting conditions, orientation, occlusion or the object's own changes in appearance. Due to these problems, multiple object tracking is still a challenging task. For a comprehensive, recent overview of the problem, see the literature review at [3].

The most common multi-object tracking targets are pedestrians, faces and vehicles.

In this thesis, I am going to tackle tracking vehicles in road traffic footages.

4.2 Metrics

The MOT challenge¹ is one of the most popular currently used multiple object tracking benchmarks. Its most recent version is the MOT20 benchmark.

4.3 Solutions

Most of the leading, publicly-disclosed multiple object tracking methods are detection-based, and thus are not end-to-end, as the system is not trained in one go, but for subtasks and then assembled later.

4.4 My measurement

4.4.1 Data

I will evaluate the model's performance for multi-object tracking on the UA-DETRAC dataset². The dataset contains 100 videos (60 for training, 40 for testing) of road traffic captured at different locations in China. The total length of the video footage is around 10 hours, stored frame by frame (as 960 pixel by 540 pixel images), at the rate of 25 frames per second.

The annotations contain information about vehicle type, illumination, scale (proportional to the square root of the bounding box area), occlusion ratio (the measure by which other objects occlude the vehicle) and truncation ratio (the degree of the bounding box lying outside the frame).

4.4.2 Benchmark

The dataset and the benchmark is described in [5]. The article also proposes an evaluation protocol for multi-object tracking. A key point is the joint analysis of detection and tracking performance, analysing the effects of the chosen model's precision/recall values (and the underlying confidence threshold setting that influences both) in relation with the tracking performance, as reflected by the MOTA and MOTP score. These relationships are visualized on the PR-MOTA and PR-MOTP curves **TODO: visualization**.

The authors argue that, as it is not fair, nor enough to compare the performance of two object detectors based on different points on the PR curve, it is also not enough to determine the maximum point on the PR-MOTA curve, as a good tracker must produce good scores in a wider range of settings. The whole range of the curve must be taken into account in some form, thus the need for a new metric:

$$\Omega^* = \frac{1}{2} \int_C \Psi(p, r) d\mathbf{s}$$

where Ψ is the MOTA score across the whole dataset at precision p and recall r , and we calculate the (approximate value) of the area under the PR-MOTA curve as an integral

¹<https://motchallenge.net/>

²<https://detrac-db.rit.albany.edu/>

along the PR curve \mathcal{C} (for every $(p, r) \in \mathcal{C}$). Similar metrics can be defined for the MOTP, FP, FN, IDS, MT and ML scores.

TODO: explain MOTA, MOTP, FP, FN, IDS, MT, ML.

Usually, a baseline for detection performance in MOT is the R-CNN architecture and its variants.

For the MOT task, I have chosen the SORT architecture, introduced in [1]. Although not the most recent, it is the architecture I have examined in my project laboratory, and has the analytical advantage of solely relying on detection performance, as opposed to methods like DeepSORT that are influenced by the association method as well. It can be considered a reasonable baseline method, relying on first-order velocity estimation and smoothing measurement errors based on the Kalman Filter.

Chapter 5

A L^AT_EX-sablon használata

Ebben a fejezetben röviden, implicit módon bemutatjuk a sablon használatának módját, ami azt jelenti, hogy sablon használata ennek a dokumentumnak a forráskódját tanulmányozva válik teljesen világossá. Amennyiben a szoftver-kéretrendszer telepítve van, a sablon alkalmazása és a dolgozat szerkesztése L^AT_EX-ben a sablon segítségével tapasztalataink szerint jóval hatékonyabb, mint egy WYSWYG (*What You See is What You Get*) típusú szövegszerkesztő esetén (pl. Microsoft Word, OpenOffice).

5.1 Címkék és hivatkozások

A L^AT_EX dokumentumban címkéket (`\label`) rendelhetünk ábrákhoz, táblázatokhoz, fejezetekhez, listákhoz, képletekhez stb. Ezekre a dokumentum bármely részében hivatkozhatunk, a hivatkozások automatikusan feloldásra kerülnek.

A sablonban makrókat definiáltunk a hivatkozások megkönnyítéséhez. Ennek megfelelően minden ábra (*figure*) címkeje `fig:` kulcsszóval kezdődik, míg minden táblázat (*table*), képlet (*equation*), fejezet (*section*) és lista (*listing*) rendre a `tab:, eq:, sec:` és `lst:` kulcsszóval kezdődik, és a kulcsszavak után tetszőlegesen választott címke használható. Ha ezt a konvenciót betartjuk, akkor az előbbi objektumok számára rendre a `\figref`, `\tabref`, `\eqref`, `\sectref` és `\listref` makrókkal hivatkozhatunk. A makrók paramétere a címke, amelyre hivatkozunk (a kulcsszó nélkül). Az összes említett hivatkozástípus, beleértve az `\url` kulcsszóval bevezetett web-hivatkozásokat is a `hyperref`¹ csomagnak köszönhetően aktívak a legtöbb PDF-nézegetőben, rájuk kattintva a dokumentum megfelelő oldalára ugrik a PDF-néző vagy a megfelelő linket megnyitja az alapértelmezett böngészővel. A `hyperref` csomag a kimeneti PDF-dokumentumba könyvjelzőket is készít a tartalomjegyzékből. Ez egy szintén aktív tartalomjegyzék, amelynek elemeire kattintva a nézegető behozza a kiválasztott fejezetet.

5.2 Ábrák és táblázatok

Használunk vektorgrafikus ábrákat, ha van rá módunk. PDFLaTeX használata esetén PDF formátumú ábrákat lehet beilleszteni könnyen, az EPS (PostScript) vektorgrafikus képformátum beillesztését a PDFLaTeX közvetlenül nem támogatja (de lehet konvertálni,

¹Segítségével a dokumentumban megjelenő hivatkozások nem csak dinamikussá válnak, de színezhetők is, bővebbet erről a csomag dokumentációjában találunk. Ez egyúttal egy példa lábjegyzet írására.

lásd később). Ha vektorgrafikus formában nem áll rendelkezésünkre az ábra, akkor a veszteségmentes PNG, valamint a veszteséges JPEG formátumban érdemes elmenteni. Figyeljünk arra, hogy ilyenkor a képek felbontása elég nagy legyen ahhoz, hogy nyomtatásban is megfelelő minőséget nyújtson (legalább 300 dpi javasolt). A dokumentumban felhasznált képfájlokat a dokumentum forrása mellett érdemes tartani, archiválni, mivel ezek hiányában a dokumentum nem fordul újra. Ha lehet, a vektorgrafikus képeket vektorgrafikus formátumban is érdemes elmenteni az újrafelhasználhatóság (az átszerkeszthetőség) érdekében.

Kapcsolási rajzok legtöbbször kimásolhatók egy vektorgrafikus programba (pl. CorelDraw) és onnan nagyobb felbontással raszterizálva kimenthetők PNG formátumban. Ugyanakkor kiválasztott ábrák készíthetők Microsoft Visio vagy hasonló program használatával is: Visio-ból az ábrák közvetlenül PDF-be is menthetők.

Lehetőségeink Matlab ábrák esetén:

- Képernyőlopás (*screenshot*) is elfogadható minőségű lehet a dokumentumban, de általában jobb felbontást is el lehet érni más módszerrel.
- A Matlab ábrát a *File/Save As* opcióval lementhetjük PNG formátumban (ugyanaz itt is érvényes, mint korábban, ezért nem javasoltuk).
- A Matlab ábrát az *Edit/Copy figure* opcióval kimásolhatjuk egy vektorgrafikus programba is és onnan nagyobb felbontással raszterizálva kimenthetjük PNG formátumban (nem javasolt).
- Javasolt megoldás: az ábrát a *File/Save As* opcióval EPS *vektorgrafikus* formátumban elmentjük, PDF-be konvertálva beillesztjük a dolgozatba.

Az EPS kép az epstopdf programmal² konvertálható PDF formátumba. Célszerű egy batch-fájlt készíteni az összes EPS ábra lefordítására az alábbi módon (ez Windows alatt működik).

```
@echo off
for %%j in (*.eps) do (
    echo converting file "%%j"
    epstopdf "%%j"
)
echo done .
```

Egy ilyen parancsfájlt (convert.cmd) elhelyeztük a sablon *figures\eps* könyvtárába, így a felhasználónak csak annyi a dolga, hogy a *figures\eps* könyvtárba kimenti az EPS formátumú vektorgrafikus képet, majd lefuttatja a convert.cmd parancsfájlt, ami PDF-be konvertálja az EPS fájlt.

Ezek után a PDF-ábrát ugyanúgy lehet a dokumentumba beilleszteni, mint a PNG-t vagy a JPEG-et. A megoldás előnye, hogy a lefordított dokumentumban is vektorgrafikusan tárolódik az ábra, így a mérete jóval kisebb, mintha raszterizáltuk volna beillesztés előtt. Ez a módszer minden – az EPS formátumot ismerő – vektorgrafikus program (pl. CorelDraw) esetén is használható.

A képek beillesztésére a ??+ ??ben mutattunk be példát (??+ ??). Az előző mondatban egyúttal az automatikusan feloldódó ábrahivatkozásra is láthatunk példát. Több képfájlt is beilleszthetünk egyetlen ábrába. Az egyes képek közötti horizontális és vertikális margót metrikusan szabályozhatjuk (figure 5.1). Az ábrák elhelyezését számtalan tipográfiai szabály egyidejű teljesítésével a fordító maga végzi, a dokumentum írója csak preferenciáit

²a korábban említett L^AT_EX-disztribúciókban megtalálható

jelezheti a fordító felé (olykor ez bosszúságot is okozhat, ilyenkor pl. a kép méretével lehet játszani).

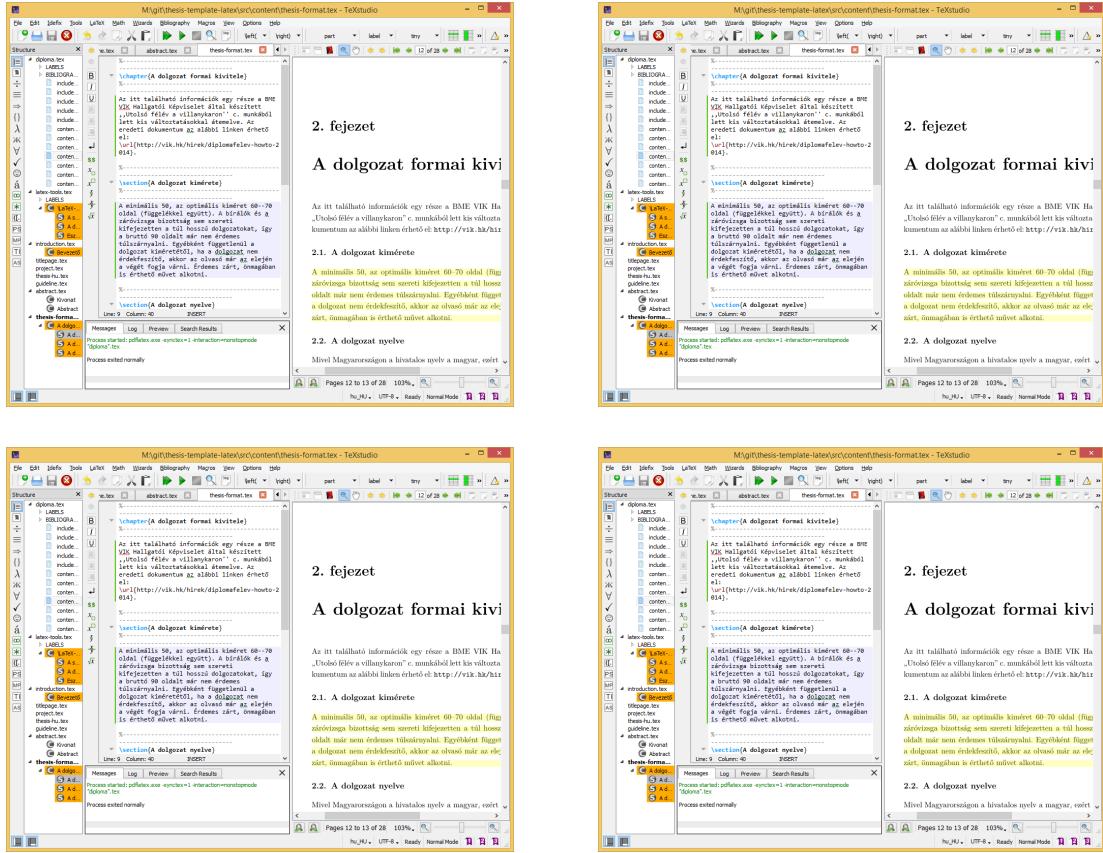


Figure 5.1: Több képfájl beillesztése esetén térközöket is érdemes használni.

A táblázatok használatára az 5.1 táblázat mutat példát. A táblázatok formázásához hasznos tanácsokat találunk a booktabs csomag dokumentációjában.

Órajel	Frekvencia	Cél pin
CLKA	100 MHz	FPGA CLK0
CLKB	48 MHz	FPGA CLK1
CLKC	20 MHz	Processor
CLKD	25 MHz	Ethernet chip
CLKE	72 MHz	FPGA CLK2
XBUF	20 MHz	FPGA CLK3

Table 5.1: Az órajel-generátor chip órajel-kimenetei.

5.3 Felsorolások és listák

Számosztan felsorolásra mutat példát a jelenlegi bekezdés:

- *első bajusz*: ide lehetne írni az első elem kifejését,
- *második bajusz*: ide lehetne írni a második elem kifejését,

- *ez meg egy szakáll*: ide lehetne írni a harmadik elem kifejését.

Számoszt felsorolást is készíthetünk az alábbi módon:

1. *első bajusz*: ide lehetne írni az első elem kifejését, és ez a kifejtés így néz ki, ha több sorosra sikeredik,
2. *második bajusz*: ide lehetne írni a második elem kifejését,
3. *ez meg egy szakáll*: ide lehetne írni a harmadik elem kifejését.

A felsorolásokban sorok végén vessző, az utolsó sor végén pedig pont a szokásos írásjel. Ez alól kivételt képezhet, ha az egyes elemek több teljes mondatot tartalmaznak.

Listákban a dolgozat szövegétől elkülönítendő kódrészleteket, programsorokat, pszeudo-kódokat jeleníthetünk meg (5.1. kódrészlet).

```
\begin{enumerate}
\item \emph{első bajusz}: ide lehetne írni az első elem kifejését,
    és ez a kifejtés így néz ki, ha több sorosra sikeredik,
\item \emph{második bajusz}: ide lehetne írni a második elem kifejését,
\item \emph{ez meg egy szakáll}: ide lehetne írni a harmadik elem kifejését.
\end{enumerate}
```

Listing 5.1: A fenti számoszt felsorolás L^AT_EX-forráskódja

A lista keretét, háttérszínét, egész stílusát megválaszthatjuk. Ráadásul különféle programnyelveket és a nyelveken belül kulcsszavakat is definiálhatunk, ha szükséges. Erről bővebbet a *listings* csomag hivatalos leírásában találhatunk.

5.4 Képletek

Ha egy formula nem túlságosan hosszú, és nem akarjuk hivatkozni a szövegből, mint például a $e^{i\pi} + 1 = 0$ képlet, *szövegközi képletként* szokás leírni. Csak, hogy másik példát is lássunk, az $U_i = -d\Phi/dt$ Faraday-törvény a $\text{rot } E = -\frac{dB}{dt}$ differenciális alakban adott Maxwell-egyenlet felületre vett integráljából vezethető le. Látható, hogy a L^AT_EX-fordító a sorközöket betartja, így a szöveg szedése esztétikus marad szövegközi képletek használata esetén is.

Képletek esetén az általános konvenció, hogy a kis félkövér betűk (**v**) oszlopvektort – és ennek megfelelően \mathbf{v}^T sorvektort – a kapitalis félkövér betűk (**V**) mátrixot jelölnek. Ha ettől el szeretnénk térdílni, akkor az alkalmazni kívánt jelölésmódot célszerű külön alfejezetben definiálni. Ennek megfelelően, amennyiben **y** jelöli a mérések vektorát, ϑ a paraméterek vektorát és $\hat{\mathbf{y}} = \mathbf{X}\vartheta$ a paraméterekben lineáris modellt, akkor a *Least-Squares* értelemben optimális paraméterbecslő $\hat{\vartheta}_{LS} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$ lesz.

Emellett kiemelt, sorszámoszt képleteket is megadhatunk, ennél az *equation* és a *eqnarray* környezetek helyett a korszerűbb *align* környezet alkalmazását javasoljuk (több okból, különféle problémák elkerülése végett, amelyekre most nem térünk ki). Tehát

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}, \quad (5.1)$$

$$\mathbf{y} = \mathbf{C}\mathbf{x}, \quad (5.2)$$

ahol **x** az állapotvektor, **y** a mérések vektora és **A**, **B** és **C** a rendszert leíró paramétermátrixok. Figyeljük meg, hogy a két egyenletben az egyenlőségjelek egymáshoz igazítva

jelennek meg, mivel a mindenkettőt az & karakter előzi meg a kódban. Lehetőség van számosztalán kiemelt képlet használatára is, például

$$\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{Bu}, \\ \mathbf{y} = \mathbf{Cx}.$$

Mátrixok felírására az $\mathbf{Ax} = \mathbf{b}$ inhomogén lineáris egyenlet részletes kifejtésével mutatunk példát:

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}. \quad (5.3)$$

A \frac utasítás hatékonyságát egy általános másodfokú tag átviteli függvényén keresztül mutatjuk be, azaz

$$W(s) = \frac{A}{1 + 2T\xi s + s^2 T^2}. \quad (5.4)$$

A matematikai mód minden szimbólumának és képességének a bemutatására termézetesen itt nincs lehetőség, de gyors referenciaként hatékonyan használhatók a következő linkek:
http://www.artofproblemsolving.com/LaTeX/AoPS_L_GuideSym.php,
<http://www.ctan.org/tex-archive/info/symbols/comprehensive/symbols-a4.pdf>,
<ftp://ftp.ams.org/pub/tex/doc/amsmath/short-math-guide.pdf>.

Ez pedig itt egy magyarázat, hogy miért érdemes align környezetet használni:
<http://texblog.net/latex-archive/math/eqnarray-align-environment/>.

5.5 Irodalmi hivatkozások

Egy L^AT_EX dokumentumban az irodalmi hivatkozások definíciójának két módja van. Az egyik a \thebibliography környezet használata a dokumentum végén, az \end{document} lezárást előtt.

```
\begin{thebibliography}{9}
\bibitem{Lamport94} Leslie Lamport, \emph{\LaTeX: A Document Preparation System}. Addison Wesley, Massachusetts, 2nd Edition, 1994.
\end{thebibliography}
```

Ezek után a dokumentumban a \cite{Lamport94} utasítással hivatkozhatunk a forrásra. A fenti megadás viszonylag kötetlen, a szerző maga formázza az irodalomjegyzéket (ami gyakran inkonzisztens eredményhez vezet).

Egy sokkal professzionálisabb módszer a BiB^LT_EX használata, ezért ez a sablon is ezt támogatja. Ebben az esetben egy külön szöveges adatbázisban definiáljuk a forrásmunkákat, és egy külön stílusfájl határozza meg az irodalomjegyzék kinézetét. Ez, összhangban azzal, hogy külön formátumkonvenció határozza meg a folyóirat-, a könyv-, a konferenciacikk-stb. hivatkozások kinézetét az irodalomjegyzékben (a sablon használata esetén ezzel nem is kell foglalkoznia a hallgatónak, de az eredményt célszerű ellenőrizni). felhasznált hivatkozások adatbázisa egy .bib kiterjesztésű szöveges fájl, amelynek szerkezetét a Az 5.2 kód részlet demonstrálja. A forrásmunkák bevitelkor a sor végi vesszők külön figyelmet igényelnek, mert hiányuk a BiB^LT_EX-fordító hibaüzenetét eredményezi. A forrásmunkákat

típus szerinti kulcsszó vezeti be (@book könyv, @inproceedings konferenciakiadványban megjelent cikk, @article folyóiratban megjelent cikk, @techreport valamelyik egyetem gondozásában megjelent műszaki tanulmány, @manual műszaki dokumentáció esetén stb.). Nemcsak a megjelenés stílusa, de a kötelezően megadandó mezők is típusról-típusra változnak. Egy jól használható referencia a <http://en.wikipedia.org/wiki/BibTeX> oldalon található.

```

@book{Wettl04,
  author      = {Ferenc Wettl and Gyula Mayer and Péter Szabó},
  publisher   = {Panem Könyvkiadó},
  title       = {\LaTeX-kézikönyv},
  year        = {2004},
}

@article{Candy86,
  author      = {James C. Candy},
  journaltitle = {{IEEE} Trans.\ on Communications},
  month       = {01},
  note        = {\doi{10.1109/TCOM.1986.1096432}},
  number      = {1},
  pages       = {72--76},
  title       = {Decimation for Sigma Delta Modulation},
  volume      = {34},
  year        = {1986},
}

@inproceedings{Lee87,
  author      = {Wai L. Lee and Charles G. Sodini},
  booktitle   = {Proc.\ of the IEEE International Symposium on Circuits and Systems},
  location    = {Philadelphia, PA, USA},
  month       = {05-4--7},
  pages       = {459--462},
  title       = {A Topology for Higher Order Interpolative Coders},
  vol         = {2},
  year        = {1987},
}

@thesis{KissPhD,
  author      = {Peter Kiss},
  institution = {Technical University of Timi\c{s}oara, Romania},
  month       = {04},
  title       = {Adaptive Digital Compensation of Analog Circuit Imperfections for Cascaded Delta-Sigma Analog-to-Digital Converters},
  type        = {phdthesis},
  year        = {2000},
}

@manual{Schreier00,
  author      = {Richard Schreier},
  month       = {01},
  note        = {\url{http://www.mathworks.com/matlabcentral/fileexchange/}},
  organization = {Oregon State University},
  title       = {The Delta-Sigma Toolbox v5.2},
  year        = {2000},
}

@misc{DipPortal,
  author      = {{Budapesti Műszaki és Gazdaság tudományi Egyetem Villamosmérnöki és Informatikai Kar}},
  howpublished = {\url{http://diplomaterv.vik.bme.hu/}},
  title       = {Diplomaterv portál (2011. február 26.)},
}

@incollection{Mkrtychev:1997,
  author      = {Mkrtychev, Alexey},
  booktitle   = {Logical Foundations of Computer Science},
  doi         = {10.1007/3-540-63045-7_27},
  editor      = {Adian, Sergei and Nerode, Anil},
  isbn        = {978-3-540-63045-6},
  pages       = {266-275},
  publisher   = {Springer Berlin Heidelberg},
}

```

```

series      = {Lecture Notes in Computer Science},
title       = {Models for the logic of proofs},
url         = {http://dx.doi.org/10.1007/3-540-63045-7_27},
volume     = {1234},
year        = {1997},
}

```

Listing 5.2: Példa szöveges irodalomjegyzék-adatbázisra BibTeX használata esetén.

A stílusfájl egy .sty kiterjesztésű fájl, de ezzel lényegében nem kell foglalkozni, mert vannak beépített stílusok, amelyek jól használhatók. Ez a sablon a BiBTeX-et használja, a hozzá tartozó adatbázisfájl a mybib.bib fájl. Megfigyelhető, hogy az irodalomjegyzéket a dokumentum végére (a \end{document} utasítás elő) beillesztett \bibliography{mybib} utasítással hozhatjuk létre, a stílusát pedig ugyanitt a \bibliographystyle{plain} utasítással adhatjuk meg. Ebben az esetben a plain előre definiált stílust használjuk (a sablonban is ezt állítottuk be). A plain stíluson kívül természetesen számtalan más előre definiált stílus is létezik. Mivel a .bib adatbázisban ezeket megadtuk, a BiBTeX-fordító is meg tudja különböztetni a szerzőt a címtől és a kiadótól, és ez alapján automatikusan generálódik az irodalomjegyzék a stílusfájl által meghatározott stílusban.

Az egyes forrásmunkákra a szövegből továbbra is a \cite parancsral tudunk hivatkozni, így az 5.2. kód részlet esetén a hivatkozások rendre \cite{Wettl04}, \cite{Candy86}, \cite{Lee87}, \cite{KissPhD}, \cite{Schreirer00}, \cite{Mkrtychev:1997} és \cite{DipPortal}. Az egyes forrásmunkák sorszáma az irodalomjegyzék bővítményekor változhat. Amennyiben az aktuális számhoz illeszkedő névelőt szeretnénk használni, használjuk az \acite{} parancsot.

Az irodalomjegyzékben alapértelmezésben csak azok a forrásmunkák jelennek meg, amelyekre található hivatkozás a szövegben, és ez így alapvetően helyes is, hiszen olyan forrásmunkákat nem illik az irodalomjegyzékbe írni, amelyekre nincs hivatkozás.

Mivel a fordítási folyamat során több lépésben oldódnak fel a szimbólumok, ezért gyakran többször is le kell fordítani a dokumentumot. Ilyenkor ez első 1-2 fordítás esetleg szimbólum-feloldásra vonatkozó figyelmeztető üzenettel zárul. Ha hibaüzenettel zárul bármelyik fordítás, akkor nincs értelme megismételni, hanem a hibát kell megkeresni. A .bib fájl megváltoztatáskor sokszor nincs hatása a változtatásnak azonnal, mivel nem minden fut újra a BibTeX fordító. Ezért célszerű a változtatás után azt manuálisan is lefuttatni (TeXstudio esetén Tools/Bibliography).

Hogy a szövegbe ágyazott hivatkozások kinézetét demonstráljuk, itt most sorban meghivatkozzuk a [?], [?], [?], [?], [?] és a [?]³ forrásmunkát, valamint a [?] weboldalt.

Megjegyzendő, hogy az ékezes magyar betűket is tartalmazó .bib fájl az inputenc csomaggal betöltött latin2 betűkészlet miatt fordítható. Ugyanez a .bib fájl hibaüzenettel fordul egy olyan dokumentumban, ami nem tartalmazza a \usepackage[latin2]{inputenc} sort. Speciális igény esetén az irodalmi adatbázis általánosabb érvényűvé tehető, ha az ékezes betűket speciális latex karakterekkel helyettesítjük a .bib fájlban, pl. á helyett \'{a}-t vagy ö helyett \H{o}-t írunk.

³Informatikai témában gyakran hivatkozunk cikkeket a Springer LNCS valamely kötetéből, ez a hivatkozás erre mutat egy helyes példát.

Irodalomhivatkozásokat célszerű először olyan szolgáltatásokban keresni, ahol jó minőségű bejegyzések találhatók (pl. ACM Digital Library,⁴ DBLP,⁵ IEEE Xplore,⁶ SpringerLink⁷) és csak ezek után használni kevésbé válogatott forrásokat (pl. Google Scholar⁸). A jó minőségű bejegyzéseket is érdemes megfelelően tisztítani.⁹ A sablon angol nyelvű változatában használt plainnat beállítás egyik sajátossága, hogy a cikkhez generált hivatkozás a cikk DOI-ját és URL-jét is tartalmazza, ami gyakran duplikátumhoz vezet – érdemes tehát a DOI-kat tartalmazó URL mezőket törölni.

5.6 A dolgozat szerkezete és a forrásfájlok

A diplomatervsablonban a TeX fájlok két alkönyvtárban helyezkednek el. Az include könyvtárban azok szerepelnek, amiket tipikusan nem kell szerkesztenünk, ezek a sablon részei (pl. cíboldal). A content alkönyvtárban pedig a saját munkánkat helyezhetjük el. Itt érdemes az egyes fejezeteket külön TeX állományokba rakni.

A diplomatervsablon (a kari irányelvek szerint) az alábbi fő fejezetekből áll:

1. 1 oldalas *tájékoztató* a szakdolgozat/diplomaterv szerkezetéről (include/guideline.tex), ami a végső dolgozatból törlendő,
2. *feladatkiírás* (include/project.tex), a dolgozat nyomtatott verzójában ennek a helyére kerül a tanszék által kiadott, a tanszékvezető által aláírt feladatkiírás, a dolgozat elektronikus verziójába pedig a feladatkiírás egyáltalán ne kerüljön bele, azt külön tölti fel a tanszék a diplomaterv-honlapra,
3. *cíboldal* (include/titlepage.tex),
4. *tartalomjegyzék* (thesis.tex),
5. a diplomatervező *nyilatkozata* az önálló munkáról (include/declaration.tex),
6. 1-2 oldalas tartalmi *összefoglaló* magyarul és angolul, illetve elkészíthető még további nyelveken is (content/abstract.tex),
7. *bevezetés*: a feladat értelmezése, a tervezés célja, a feladat indokoltsága, a diplomaterv felépítésének rövid összefoglalása (content/introduction.tex),
8. sorszámmal ellátott *fejezetek*: a feladatkiírás pontosítása és részletes elemzése, előzmények (irodalomkutatás, hasonló alkotások), az ezekből levonható következtetések, a tervezés részletes leírása, a döntési lehetőségek értékelése és a választott megoldások indoklása, a meghalászott műszaki alkotás értékelése, kritikai elemzése, továbbfejlesztési lehetőségek,
9. esetleges *köszönetnyilvánítások* (content/acknowledgement.tex),
10. részletes és pontos *irodalomjegyzék* (ez a sablon esetében automatikusan generálódik a thesis.tex fájlban elhelyezett \bibliography utasítás hatására, az section 5.5ban leírtak szerint),

⁴<https://dl.acm.org/>

⁵<http://dblp.uni-trier.de/>

⁶<http://ieeexplore.ieee.org/>

⁷<https://link.springer.com/>

⁸<http://scholar.google.com/>

⁹<https://github.com/FTSRG/cheat-sheets/wiki/BibTeX-Fixing-entries-from-common-sources>

11. függelékek (content/appendices.tex).

A sablonban a fejezetek a thesis.tex fájlba vannak beillesztve \include utasítások segítségével. Lehetőség van arra, hogy csak az éppen szerkesztés alatt álló .tex fájlt fordítsuk le, ezzel lerövidítve a fordítási folyamatot. Ezt a lehetőséget az alábbi kódrészlet biztosítja a thesis.tex fájlban.

```
\includeonly{  
    guideline,  
    project,  
    titlepage,  
    declaration,  
    abstract,  
    introduction,  
    chapter1,  
    chapter2,  
    chapter3,  
    acknowledgement,  
    appendices,  
}
```

Ha az alábbi kódrészletben az egyes sorokat a % szimbólummal kikommentezzük, akkor a megfelelő .tex fájl nem fordul le. Az oldalszámok és a tartalomjegyék természetesen csak akkor billennek helyre, ha a teljes dokumentumot lefordítjuk.

5.7 Alapadatok megadása

A diplomaterv alapadatait (cím, szerző, konzulens, konzulens titulusa) a `thesis.tex` fájlban lehet megadni.

5.8 Új fejezet írása

A főfejezetek külön content könyvtárban foglalnak helyet. A sablonhoz 3 fejezet készült. További főfejezeteket úgy hozhatunk létre, ha új TeX fájlt készítünk a fejezet számára, és a `thesis.tex` fájlban, a `\include` és `\includeonly` utasítások argumentumába felvesszük az új `.tex` fájl nevét.

5.9 Definíciók, tételek, példák

Definition 1 (Fluxuskondenzátor térerőssége). Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

Example 1. Példa egy példára. *Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.*

Theorem 1 (Kovács tétele). *Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.*

Bibliography

- [1] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE, sep 2016. DOI: 10.1109/icip.2016.7533003. URL <https://doi.org/10.1109/2Ficip.2016.7533003>.
- [2] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2014. URL <https://arxiv.org/abs/1405.0312>.
- [3] Wenhan Luo, Junliang Xing, Anton Milan, Xiaoqin Zhang, Wei Liu, and Tae-Kyun Kim. Multiple object tracking: A literature review. *Artificial Intelligence*, 293:103448, apr 2021. DOI: 10.1016/j.artint.2020.103448. URL <https://doi.org/10.1016%2Fj.artint.2020.103448>.
- [4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL <https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fdbd053c1c4a845aa-Paper.pdf>.
- [5] Longyin Wen, Dawei Du, Zhaowei Cai, Zhen Lei, Ming-Ching Chang, Honggang Qi, Jongwoo Lim, Ming-Hsuan Yang, and Siwei Lyu. UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking. *Computer Vision and Image Understanding*, 2020.