

3D-positiebepaling van een persoon in een leefruimte

Jeroen STREULENS

Promotor: prof. dr. ir. Toon Goedemé

Co-promotor: ing. Timothy Callemein

Masterproef ingediend tot het behalen van
de graad van master of Science in de
industriële wetenschappen: Elektronica-ict
ICT

©Copyright KU Leuven

Zonder voorafgaande schriftelijke toestemming van zowel de promotor(en) als de auteur(s) is overnemen, kopiëren, gebruiken of realiseren van deze uitgave of gedeelten ervan verboden. Voor aanvragen i.v.m. het overnemen en/of gebruik en/of realisatie van gedeelten uit deze publicatie, kan u zich richten tot KU Leuven Technologiecampus De Nayer, Jan De Nayerlaan 5, B-2860 Sint-Katelijne-Waver, +32 15 31 69 44 of via e-mail iiw.denayer@kuleuven.be.

Voorafgaande schriftelijke toestemming van de promotor(en) is eveneens vereist voor het aanwenden van de in deze masterproef beschreven (originele) methoden, producten, schakelingen en programma's voor industrieel of commercieel nut en voor de inzending van deze publicatie ter deelname aan wetenschappelijke prijzen of wedstrijden.

Dankwoord

Een thesis is het grote project dat de afsluiting vormt van een opleiding. Een jaar lang onderzoeken, nadelen, stressen en nog meer onderzoeken en dit met maar één doel: het masterdiploma halen. Uiteraard zou het resultaat dat ik vandaag kan voorleggen niet tot stand zijn gekomen zonder een heel aantal andere mensen.

Daarom zou ik zeker mijn dank willen uitbrengen aan prof. dr. ir. Toon Goedemé en ing. Timothy Callemein om een onderwerp aan te bieden dat in mijn interessegebied ligt. Ik wil hen ook bedanken voor de feedback, de vele antwoorden die ze mij gegeven hebben op al mijn vragen en de goede begeleiding die ik het afgelopen jaar gekregen heb. Verder wil ik ook de onderzoeksgroep EAVISE bedanken voor alle extra hulp die ik gekregen heb de afgelopen jaren.

Ook wil ik mijn medestudenten bedanken. Het afgelopen jaar konden we bij elkaar terecht om elkaar te motiveren om zo tot het einde door te gaan. Daarnaast zou ik zeker en vast mijn familie willen bedanken. Zij hielpen bij het nalezen van deze thesis, verdroegen de stress die ik op sommige momenten moest doorstaan en motiveerden me op de juiste momenten. In het bijzonder wil ik mijn vriendin Jana bedanken om mijn thesis niet één, niet twee, maar een heel aantal keren te herlezen en te herschrijven. Zonder haar zou deze masterproef dan ook veel meer schrijffouten bevatten. Zij bood me steun op de nodige momenten en wist me altijd te motiveren.

Zonder al deze mensen zou dit resultaat er zeker niet zijn.

Jeroen Streulens

Abstract

Deze thesis heeft als doel het benaderen van de top-down positie van een persoon in een leefruimte aan de hand van statische camera's zonder diepte-informatie. In een eerste fase worden een aantal technieken met elkaar vergeleken die we voor dit onderzoek konden gebruiken. Zo worden er technieken besproken voor het detecteren van mensen op camerabeelden, het kalibreren van camera's en het bepalen van de 3D-pose van een persoon aan de hand van één of meerdere beelden.

Op basis van deze literatuurstudie hebben we een aantal technieken gekozen die we verder uitwerkten. Zo hebben we onderzocht of VNect een bruikbare techniek is voor het bepalen van de 3D-pose van de persoon. Aangezien deze techniek niet occlusie-ongevoelig is en occlusie in een leefruimte een veelvoorkomend fenomeen is, zijn we hier niet dieper op ingegaan. We kozen daar tegenover het OpenPose framework om de 2D-pose van de persoon op de camerabeelden te benaderen. Verder hebben we een implementatie van het five-point pose algoritme getest om de verschillende camera's te kalibreren. Deze implementatie werkte echter niet, waardoor we zelf een techniek hebben uitgewerkt die we later geïmplementeerd hebben. Bij deze implementatie kan de gebruiker de externe kalibratieparameters manueel bepalen door bepaalde punten in de cameraviews en de overeenkomstige punten op de kaart van de leefruimte aan te klikken. Verder hebben we ook zelf een implementatie geschreven gebaseerd op het Monocular Depth Perception algoritme. Deze implementatie vormt projectielijnen op het grondplan, tussen de camera en de persoon, en dit op basis van de pose die verkregen wordt aan de hand van OpenPose. De positie van de persoon wordt dan benaderd door het snijpunt van de projectielijnen van verschillende camera's.

Het resultaat van deze thesis is een systeem dat de positie van de persoon zo goed mogelijk zal benaderen indien de persoon zichtbaar is in één of meerdere views. Validatie op eigen data en op de CMU-dataset tonen aan dat de door onze implementatie geschatte positie van de persoon gemiddeld 0.4763 meter van de werkelijke positie verwijderd is.

Keywords: top-down, positiebepaling, kalibratie, hoek-per-pixel, leefruimte

Abstract

This thesis contains research on how to approach the top-down position of a human in a living area, using static cameras without any depth information. After analyzing related work in this area, we selected a couple of techniques that we thought would be fit for our research.

At first, we tried out a technique called VNect. This technique approaches the 3D pose of a human using monocular images. Unfortunately we discovered that this technique does not work when the person is partly occluded. Therefore we decided to use the OpenPose framework to approach the 2D pose of the human. However, to determine the top-down position using OpenPose, we need to know the external calibration parameters of the cameras. To achieve this we tried to use Five-point pose, an algorithm that defines the external calibration parameters using corresponding points in multiple views. We adapted an implementation we found, but this implementation didn't work. To not lose time, we made our own calibration implementation in which the user can manually define the positions of the cameras by clicking on a point in a cameraview and the corresponding point on the map. Finally we made an implementation that uses the angle to pixel ratio. This approach uses these camera positions and the OpenPose output to define projection lines on the map between the camera and the person.

The result is a system that will define the position of a human as accurately as possible, given that the person is visible in one or multiple views. Validation on our own data and on the CMU dataset proves that our approach is reasonably accurate. The estimated position we compute, differs from the real position by 0.4695 meters on average, when the person is visible in two views.

Keywords: top-down, position, calibration, angle to pixel, living area

Short summary

A lot of people watch reality tv without knowing how much work it takes to make it. In order to make good reality tv, directors need a camera team at the scene 24/7, because they never know when something interesting might happen. The goal of this thesis, proposed by the research group EAVISE, is to develop a technique that approaches the 3D position of a person in a setup using static cameras without any depth information. This position will then be used to automatically aim one or more pan-tilt-zoom cameras at the person, so the film crew doesn't have to be present the entire time. The control of the pan-tilt-zoom cameras will not be treated in this research. Our research question is: how can we determine the 3D position of a person in a living area?

We subdivide this question in three questions:

- (i) Which method do we use to detect the person on the images?
- (ii) In which way are we going to locate the person in top-down view?
- (iii) How are we going to approach the 3D pose of the person?

State of the art

We've devided the main goal of this research into different subtasks. The first thing we had to do was detecting people on the monocular images. We compared human detection techniques and human pose estimation techniques. The difference between these two is that detection techniques will describe the position of a human on an image by an x and y coordinate and a score. Pose estimation techniques however approach the pose of a human by defining the position of the joints. A pose estimation technique gives more information and has more applications. We compared two pose estimation frameworks: Deepcut and OpenPose. These frameworks have both been tested on the MPII Human Pose Dataset. The results of this test showed that OpenPose is more accurate when there are multiple people visible on the image. Additionally, the OpenPose framework is updated frequently and there are updates that do not just approach the pose of the human, but also of the face and hands. Therefore we decided to use OpenPose in this thesis.

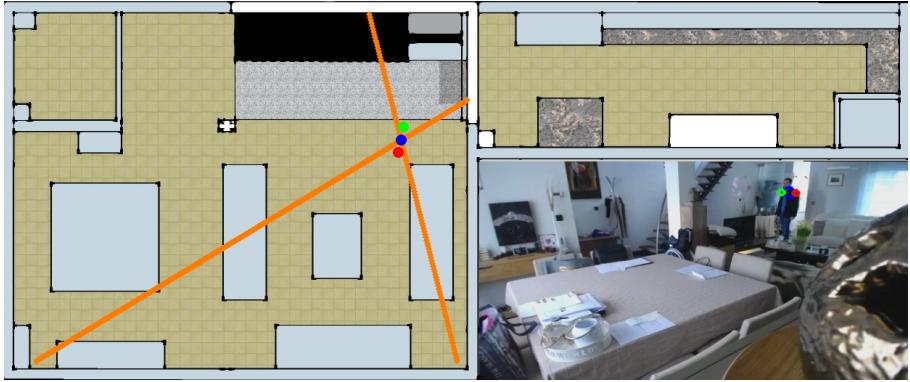
Apart from detecting the person on the images, we also need to define the external calibration parameters of the cameras. We looked into the possibility of calibration using a chessboard. The disadvantage of this technique is that the chessboard has to be simultaneously visible in two camera views. However, in a living area it is possible that the cameras are rather far apart from each other and that the overlapping area is minimal. We also wanted to create a system that can be used universally, without having to physically visit the setup. With chessboard calibration, this would be impossible. This is why we looked into a self-calibration method and the five-point pose algorithm. The five-point pose algorithm explains that it is possible to calculate the external parameters of two cameras using five point couples. These couples consist of a point on a view of the first camera and the corresponding point on a view of the other camera.

A different possible technique to define the top-down position of the human uses the angle to pixel ratio. This is a constant defined by the angle of view of a camera divided by the width of the camera resolution. If we multiply this constant with the x coordinate of a pixel, we know the angle the pixel makes with the left side of the angle of view. This way we constitute projection lines on the top-down map between the camera and the person.

Finally, we explored a couple of techniques to approach the 3D pose of a human. Two different approaches exist. A first approach uses multiple views. Examples of techniques that make use of this approach are 'Multiple human 3D pose estimation from multiview images' and 'Harvesting multiple views for marker-less 3D human pose annotations'. The disadvantage of these techniques is that when a person is not visible in multiple views, it is not possible to create a 3D pose. That's why a new approach emerged that tries to define the 3D pose of a human using only a single monocular image. Two of the most accurate techniques are '2D pose estimation + matching' and VNect. 2D pose estimation + matching uses a 2D pose estimation technique and a database. When the 2D pose is approached, it compares this pose with couples of 2D and 3D poses in the database. When there is a match with a 2D pose, the result of this technique is the coupled 3D pose. The VNect algorithm uses a convolutional neural network to estimate the 2D and 3D joint locations. The technique uses a model of a human to fit the joint locations and verify if the 3D pose is a possible pose for a human. It is a realtime algorithm that works with a speed of 33ms per frame.

Implementation

During the implementation we noticed that VNect worked very well on images without occlusion. However, when occlusion occurs, the accuracy of VNect decreases. That's why we decided not to work with VNect, but to use OpenPose as a 2D pose estimator instead. To use OpenPose, we need to know the external camera calibration parameters to know the relative position of one camera to another.



Figuur 1: Person is visible for two cameras. Orange lines are projection lines. Person is displayed by three points: green (right shoulder), blue (neck) and red (left shoulder)

To achieve this we used a public implementation of the five-point pose algorithm, but the output did not seem to be correct.

We decided to calibrate the cameras manually by searching for corresponding points between a camera view and a map of the living area. If we know these corresponding points, we know the rotation of each camera. The next step is to use the output of the OpenPose framework to determine the angle between the left side of the angle of view and the person (we chose the neck as point of orientation). This is called the 'angle per pixel method'. Using this angle, we can create a projection line on the map of the living area. When the person is visible for two or more cameras, we can approach the position of the person by the intersection of these projection lines. This is displayed in Figure 1.

Evaluation

We evaluated our approach on our own data and on 2100 frames of the CMU dataset. During the evalution, we did not use the OpenPose framework to determine the 2D pose of the person because we did not want our results to depend on the accuracy of OpenPose. That's why we projected the 3D pose to multiple 2D poses using the external calibration parameters. On the CMU dataset we approach the position with an offset of 0.4695 meters on average.

Conclusion

We conclude that our implementation approaches the position of a person in a living area with a good accuracy. We did not use VNect because of the malfunction when occlusion occurs. We did not use the five-point pose algorithm either because the implementation we found did not seem to work. Instead, we manually determined the external calibration parameters of each camera and used the angle per pixel method to determine projection lines between each camera and the person. The intersection of these projection lines approaches the position of the person. We evaluated on the CMU dataset, where we approach the position with an offset of 0.4695 meters on average.

Inhoudsopgave

1 Inleiding	1
2 Literatuurstudie	3
2.1 Detectie van mensen op 2D-beelden	3
2.1.1 Persoonsdetectie	3
2.1.2 Pose Estimation	4
2.2 Bepalen van de positie in top-down view	11
2.2.1 Kalibratie	11
2.2.2 Angle to pixel ratio	16
2.2.3 Vergelijking kalibratie en angle to pixel ratio	17
2.2.4 Probabilistic Occupancy Map	17
2.2.5 Test-datasets	18
2.3 3D-pose	19
2.3.1 Aan de hand van meerdere views	21
2.3.2 Aan de hand van single view	23
2.3.3 Sparseness Meets Deepness	25
2.3.4 2D Pose Estimation + Matching	25
2.3.5 VNect	26
3 Uitwerking	29
3.1 Specificaties	29
3.2 Pose estimation	30
3.2.1 VNect	30
3.2.2 OpenPose	33
3.3 5 Point Algorithm	37

3.4 Top-down positie	39
3.4.1 Theoretische uitwerking	39
3.4.2 Verschillende gevallen hoek-per-pixelmethode	41
3.4.3 Implementatie	43
4 Evaluatie	49
4.1 Eigen data	49
4.2 CMU-dataset	50
5 Conclusie	53
5.1 Toekomst	54

Lijst van figuren

1	Person is visible for two cameras. Orange lines are projection lines. Person is displayed by three points: green (right shoulder), blue (neck) and red (left shoulder)	xi
2.1	Voorbeeld output van een persoonsdetector, Huang et al. (2014)	4
2.2	DeepCut methode, Pishchulin et al. (2015)	7
2.3	DeeperCut output, Insafutdinov et al. (2016)	8
2.4	Part Affinity Fields methode, Cao et al. (2017)	9
2.5	Links: output volgens MPI Andriluka et al. (2014), Rechts: output volgens COCO Lin et al. (2014)	10
2.6	Self-calibration aan de hand van voetgangers, Hödlmoser and Kampel (2010)	14
2.7	θ en Φ voor positiebepaling van persoon, Hombali et al. (2011)	16
2.8	Opbouw Panoptic Studio	20
2.9	Camera types Panoptic Studio	20
2.10	Sparseness Meets Deepness overview	25
2.11	VNect overview	27
3.1	Overzicht te gebruiken technieken	29
3.2	Inputafbeeldingen van de demo van het VNect algoritme	31
3.3	Outputafbeelding van VNect met een testafbeelding als input: 2D-pose (links), heatmaps (midden) en 3D-pose(rechts)	32
3.4	Ongewenste output VNect 1	32
3.5	Ongewenste output VNect 2	33
3.6	Heatmaps horende bij Fig 3.4	34
3.7	OpenPose output	36
3.8	OpenPose fout	37

3.9 Output OpenPose JSON	38
3.10 Output 5 Point Algorithm	39
3.11 Positioneren in top-down	40
3.12 Drie camera's: niet hetzelfde snijpunt	42
3.13 Voorbeeld van een plattegrond. Rood, Geel, Groen, Blauw betekenen respectievelijk: Persoon zichtbaar op geen, één, twee of drie camera's	43
3.14 Posities van de verschillende camera's	44
3.15 Kalibreren van camera's door bepalen van puntenkoppels	45
3.16 Positie + rotatie van de verschillende camera's	45
3.17 Voorbeeld van een masker dat de muren aanduidt	46
3.18 Situatie waarin de persoon zichtbaar is voor twee camera's. Oranje lijnen zijn projectielijnen. De persoon wordt weergegeven door drie cirkels: groen (rechterschouder), blauw (nek) en rood (linkerschouder)	47
3.19 Situatie waarin de persoon zichtbaar is voor drie camera's	48
3.20 Matrix die de verschillende zones aanduidt. Rood: zichtbaar voor geen enkele camera, geel: zichtbaar voor één camera, groen: zichtbaar voor twee camera's en paars: zichtbaar voor drie of meer camera's	48
4.1 Boxplot van het verschil tussen werkelijke positie en de door onze techniek benaderde positie uitgedrukt in millimeter (2100 frames)	51

Lijst van tabellen

2.1	Vergelijking nauwkeurigheid DeeperCut en OpenPose op verschillende gewrichten. Toegepast op de Multi-Person subset van de Andriluka et al. (2014) dataset.	11
2.2	Nauwkeurigheid van de verschillende technieken op de Ionescu et al. (2014) dataset	23
4.1	Eigen data (15 frames per reeks): verschil met werkelijke positie	50

Acroniemen

AP angle to pixel ratio. 16

CNN Convolutional Neural Network. 8, 9, 21, 23–27

CPU Central Processing Unit. 10

cuDNN CUDA Deep Neural Network. 10

DPM Deformable Part Models. 3

GPU Graphics Processing Unit. 9

PAF Part Affinity Fields. 6, 8

POM Probabilistic Occupancy Map. 17, 18

PTZ pan-tilt-zoom. 1, 51

ResNet Residual Network. 7

RGB Red Green Blue. 26

RGBD Red Green Blue Depth. 18

Hoofdstuk 1

Inleiding

Veel mensen zijn fan van goede reality tv, maar voor productiehuizen is het niet altijd gemakkelijk om dit te realiseren. Indien men bijvoorbeeld 24 uur op 24 en 7 dagen op 7 wil filmen, moet men een heel team voorzien. Dit team moet dag en nacht ter plaatse zijn, kost geld en zal bovendien soms een interessant taferelen niet opmerken of er traag op reageren. In de huidige samenleving proberen we om de taken van een mens op zoveel mogelijk manieren te verlichten of te vergemakkelijken, in vele gevallen door het gebruik van technologie. Dit is zeker en vast ook mogelijk voor reality tv.

Het doel van dit onderzoek is om pan-tilt-zoom (PTZ) camera's¹ op intelligente plaatsen te bevestigen om geleidelijk aan de taak van een cameraman gedeeltelijk over te nemen. Op de PTZ camera's worden ondersteunende camera's geplaatst die een volledig beeld geven van de scène (onafhankelijk van de huidige positie van de PTZ). Op basis van meerdere scènecamera's zullen we de 3D-positie van een persoon in de leefruimte bepalen. Het doel is om dan de verschillende PTZ camera's zichzelf te laten aansturen aan de hand van de verkregen positie. Zo is de kans kleiner dat er interessante taferelen gemist worden, en moet de filmcrew niet de hele tijd aanwezig zijn. Zo'n 3D-positie bestaat uit een positie in top-down view, in combinatie met een 3D-pose van de persoon. Een 3D-pose is een benadering van de houding van de verschillende ledematen van een persoon in drie dimensies. Dit kan dienen voor toepassingen zoals virtual reality.

De eerste uitdaging in dit onderzoek is dat de camera's geplaatst worden in een leefruimte. Ze kijken niet altijd neer op de mensen zoals een bewakingscamera, maar kunnen zich dus bijvoorbeeld ook op ooghoogte bevinden. Een tweede uitdaging is het feit dat de camera's ver van elkaar verwijderd kunnen zijn. We doen dit om de kostprijs te drukken, en omdat het in een leefruimte ook niet altijd mogelijk is om overal camera's te plaatsen. Er werd ook een beperking opgelegd. Zo mogen we er in dit onderzoek van uit gaan dat er zich telkens maar één persoon in de leefruimte bevindt.

¹Een camera die door de gebruiker vanop afstand te bewegen is.

De onderzoeksraag van dit onderzoek wordt als volgt geformuleerd:

1. Hoe kunnen we de 3D-positie van een persoon in een leefruimte bepalen?

Uit deze onderzoeksraag kunnen we een aantal deelvragen opstellen:

- (i) Welke methode gebruiken we om de persoon te detecteren?
- (ii) Op welke manier kunnen we de persoon lokaliseren in top-down view?
- (iii) Welke manier gebruiken we om de 3D-pose van de persoon te benaderen?

Onze literatuurstudie, beschreven in hoofdstuk 2, is onderverdeeld in een aantal delen die overeenkomen met de verschillende deelvragen. We onderzoeken in 2.1 welke technieken reeds bestaan voor de positiebepaling van een persoon in een 2D-beeld. Zo zullen we het hebben over persoonsdetectie en pose estimation. Bij pose estimation gaan we iets dieper in op Deepcut en OpenPose. Vervolgens bekijken we in 2.2 welke technieken gebruikt kunnen worden om de positie van de persoon in top-down view te bepalen. Zo bespreken we verschillende vormen van kalibratie, en een techniek die de hoekverandering per pixel in de horizontale richting zal gebruiken om de positie van de persoon te bepalen. In 2.3 bekijken we welke technieken we kunnen gebruiken om de volledige 3D-pose te benaderen. We bespreken vier verschillende soorten technieken die dit probleem aanpakken. Zo hebben we deep learning, waarbij de 3D-pose rechtstreeks van de afbeelding afgeleid wordt. Vervolgens zijn er ook technieken die de 2D-pose van de persoon bepalen en dan een databank doorzoeken die combinaties van 2D- en 3D-poses bevat om zo de best passende 3D-pose te kiezen. Daarnaast kunnen we de 3D-pose ook bepalen door gebruik te maken van motion-informatie. Ten slotte bekijken we ook technieken die starten van een 3D-pose en deze pose door middel van regressie aanpassen tot de fout van deze pose met de werkelijke 3D-pose zo klein mogelijk is.

In hoofdstuk 3 bespreken we de uitwerking van de gevonden technieken. Zo bespreken we de implementatie van Vnect en OpenPose voor pose estimation, het five-point pose algoritme voor kalibratie uit te voeren en tot slot bespreken we de uitwerking van een techniek gebaseerd op de hoekverandering per pixel.

Tot slot bespreken we in hoofdstuk 4 de evaluatie van onze implementatie op eigen data en op de CMU-dataset en formuleren we in hoofdstuk 5 onze conclusie.

Hoofdstuk 2

Literatuurstudie

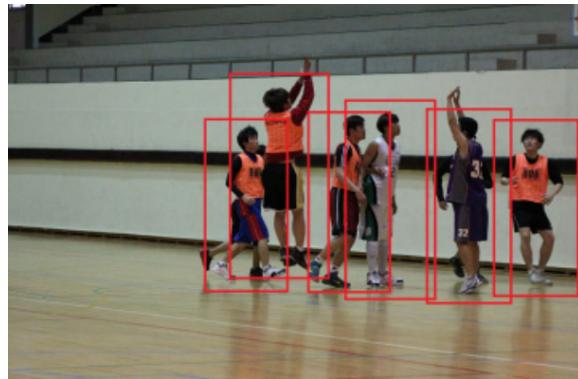
Dit onderzoek gebeurt in verschillende stappen, die vervolgens gecombineerd zullen worden. We beginnen met het zoeken naar een methode om mensen te detecteren op de 2D-beelden geleverd door de scènecamera's. Er bestaat een overvloed aan methoden, waardoor er stap voor stap moet gekeken worden welke methode geschikt is voor dit onderzoek. Het detecteren van de persoon gebeurt indien mogelijk op meerdere camera's, om vervolgens aan de hand van deze detecties zijn positie in top-down view te bepalen. Ten slotte bespreken we een aantal technieken die uit één of meerdere 2D-beelden de 3D-pose van deze persoon benaderen.

2.1 Detectie van mensen op 2D-beelden

Er bestaan reeds tal van technieken om objecten te detecteren op afbeeldingen. Bij objectdetectie is het doel om objecten van een bepaalde klasse te detecteren (Amit and Felzenszwalb (2014)). Deze klassen kunnen bijvoorbeeld auto's, fietsen of personen zijn. Indien men objecten zoekt van de klasse 'persoon', spreekt men over een persoonsdetectie. Er bestaan hiervoor twee methoden: persoonsdetectie en pose estimation.

2.1.1 Persoonsdetectie

Een persoonsdetector is een objectdetector die de objecten van de klasse 'persoon' zal detecteren. Er bestaan heel wat voorbeelden van zeer goede en krachtige persoonsdetectoren. Enkele voorbeelden hiervan zijn onder andere Viola&Jones variants (Viola et al. (2003)), Histograms of Oriented Gradients (Dalal and Triggs (2005)), Integral Channel Features (Dollar et al. (2009)), Deformable Part Models (DPM) (Felzenszwalb et al. (2010)) en Aggregate Channel Features (Yang et al. (2014)). Allemaal hanteren ze een andere methode, maar de weergave van het resultaat is hetzelfde. Elke detectie aan de hand van een objectdetector wordt aangegeven aan de hand van



Figuur 2.1: Voorbeeld output van een persoonsdetector, Huang et al. (2014)

een vorm van pose-informatie. Dit kan door middel van de locatie van het object, een locatie en een schaal of de grootte van het object aan de hand van een bounding box zoals weergegeven in Figuur 2.1. Er bestaan ook meer gedetailleerde objectdetectoren die ook de onderdelen van het gedetecteerde object weergeven. Zo zullen de locaties van de ogen, mond en neus bij de detectie van een gezicht berekend worden. Een voorbeeld van zo'n detector is Redmon and Farhadi (2016). De detectoren kunnen objecten detecteren doordat ze gebruik maken van een set van trainingsvoorbeelden. Hierbij wordt in het geval van een gezichtsdetector op een voorbeeldafbeelding een gezicht aangeduid, dit om de detector te trainen. Hoe groter de set van trainingsvoorbeelden, hoe meer objecten de detector van een bepaalde klasse zal kunnen herkennen. In deze toepassing zou een persoonsdetector voldoende zijn. Er zijn echter nieuwe technieken die niet alleen de persoon detecteren maar ook zijn pose benaderen.

2.1.2 Pose Estimation

In sectie 2.1.1 werd reeds weergegeven hoe het resultaat van een objectdetector eruitziet. Er zijn echter technieken die meer punten als output geven dan enkel de bounding box. Sigal (2014) definieert pose estimation als het proces waarbij we de configuratie/houding van het lichaam van de mens zullen benaderen aan de hand van één enkele afbeelding. De toepassingen van pose estimation zijn eindeloos. Het kan onder andere gebruikt worden voor beveiliging, persoonsherkenning, virtual reality,... Verschillende factoren bemoeilijken echter dit proces:

- Veranderlijke verschijning van mensen
- Veranderlijke belichting
- Verschillen in uiterlijk tussen mensen onderling
- Occlusie van het menselijk lichaam door voorwerpen of door andere mensen

- Complexiteit van het menselijk skelet
- Het verlies van 3D-informatie bij de omzetting naar 2D
- De vele vrijheidsgraden en gewrichten van de mens
- De situatie waarbij er twee personen gedeeltelijk voor elkaar staan

Volgens het onderzoek van Cao et al. (2017) bestaan er twee typen van multi person pose estimation technieken. Ten eerste zijn er technieken die eerst de personen op de afbeelding zullen detecteren door middel van een persoonsdetector. De volgende stap is dan om op elk gevonden persoonsoject een single-person pose estimation uit te voeren. De technieken die volgens dit principe werken worden top-down benaderingen genoemd. Het nadeel van deze werkwijze is dat men er eerst in moet slagen om de persoon te detecteren. Indien dit niet lukt, bijvoorbeeld doordat de persoon zich te dicht bij de camera bevindt of door occlusie, zal het benaderen van de pose vroegtijdig gestopt worden. Ook kan de uitvoertijd redelijk hoog oplopen, aangezien voor elke gedetecteerde persoon in de afbeelding een single-person pose estimation uitgevoerd moet worden. Een tweede mogelijkheid is het bottom-up principe. Dit type heeft het voordeel dat het geen last heeft van het vroegtijdig afbreken van de pose estimation, en dat het de mogelijkheid heeft om de runtime los te koppelen van het aantal personen op de afbeelding. Deze bottom-up technieken detecteren eerst de gewrichten van een persoon om deze vervolgens aan elkaar te koppelen en de pose van de persoon ermee te benaderen. Voor single-person pose estimation vormt dit geen enkel probleem. Indien er echter meerdere mensen in beeld zijn, kan men niet meer onderscheiden welke ledematen van welke persoon zijn. Dit probleem wordt opgelost in het onderzoek van Cao et al. (2017). We bespreken deze techniek in 2.1.2.2.

Het resultaat van pose estimation is een matrix met een set van punten die de pose van de mens benaderen. Visueel gezien krijgen we een benadering van het skelet van de mens, waarbij gewrichten en beenderen worden weergegeven door middel van bollen en lijnen. Zo geeft pose estimation een gedetailleerdere beschrijving van de gedetecteerde mens. Hoewel pose estimation nog in volle ontwikkeling is, bestaan er toch reeds veelbelovende frameworks. De bovengenoemde uitdagingen zorgen er echter voor dat er nog zeer veel mogelijkheden zijn om deze frameworks te verbeteren naar de toekomst toe. Het gebruik van pose estimation is voor dit onderzoek geen verplichting, maar geeft meer punten als output, waar verder mee kan gerekend worden in de volgende stappen. Ook geeft het een extra feature die ervoor zorgt dat er meer mogelijkheden zijn om de resultaten van dit onderzoek later te gebruiken voor verdere ontwikkeling.

In het onderzoek van Ershadi-Nasab et al. (2017) werd reeds gezocht naar de 3D-pose van meerdere mensen aan de hand van meerdere beelden. In het huidige onderzoek gaan we ervan uit dat er telkens maar één persoon in beeld is. Dit aspect van de bron mogen we dus verwaarlozen.

Het feit dat er een 3D pose estimation uitgevoerd wordt op basis van meerdere beelden is echter wel interessant. Er werd in het onderzoek van Ershadi-Nasab et al. (2017) gekozen om Deepercut als part detector te gebruiken. Er wordt wel een tweede methode vermeld, namelijk het OpenPose framework (Cao et al. (2017)), maar de auteurs geven aan dat deze techniek in sommige gevallen een verkeerde benadering van de menselijke pose zou geven. Deze opmerking wordt echter niet gestaafd met testresultaten.

Cao et al. (2017) geven in hun onderzoek meer informatie over Part Affinity Fields (PAF), een onderdeel van het OpenPose framework. In tegenstelling tot Ershadi-Nasab et al. (2017) vermelden ze hierbij dat hun methode nauwkeuriger is dan Deepercut (Insafutdinov et al. (2016)). Mogelijk kunnen we deze tegenstrijdigheid verklaren door de timing van publicatie: beide bronnen werden namelijk rond dezelfde periode geschreven, waardoor de resultaten van PAF waarschijnlijk nog niet optimaal waren toen ze door Ershadi-Nasab et al. (2017) bekijken werden. PAF zou in ons onderzoek interessanter zijn aangezien deze methode sneller lijkt te zijn, en er updates uitgevoerd zijn waarbij ook het gezicht en de handen benaderd kunnen worden. Dit kan handig zijn voor het 3D-positioneren van de mens in een leefruimte. Het is dan ook de moeite waard om beide frameworks eens meer in detail te bekijken en ze met elkaar te vergelijken, om vervolgens een beredeneerde keuze voor een framework te kunnen maken.

Er bestaan uiteraard nog meer technieken om pose estimation uit te oefenen. Zo belooft het onderzoek van Chen et al. (2017) zeer goede resultaten, maar is hun code momenteel nog niet publiek beschikbaar. Of er nog betere technieken komen op het vlak van pose estimation, kunnen we niet met zekerheid zeggen. Maar om toch met dit scenario rekening te houden, is het doel om dit onderzoek niet afhankelijk te maken van de gekozen pose estimation techniek, zodat deze later vervangen kan worden door een andere techniek.

2.1.2.1 Deepercut

Deepercut (Insafutdinov et al. (2016)) is de vernieuwde variant van DeepCut. DeepCut is een pose estimation techniek die vertrekt van een set van body part candidates (rode kruisjes in Figuur 2.2a) en body part classes. Body part candidates zijn objecten waarvan men vermoedt dat het lichaamsdelen zijn, en body part classes zijn soorten lichaamsdelen zoals bijvoorbeeld een knie, schouder of hoofd. De body part candidates worden gegenereerd door een body part detector¹, die hen elk een unary score geeft voor elke body part class. Deze score geeft de kans weer dat het lichaamsdeel tot een bepaalde body part class behoort. Aan de hand hiervan zal DeepCut kijken wat de verliezen en winsten zouden zijn voor het pose estimation probleem, indien het body part een deel

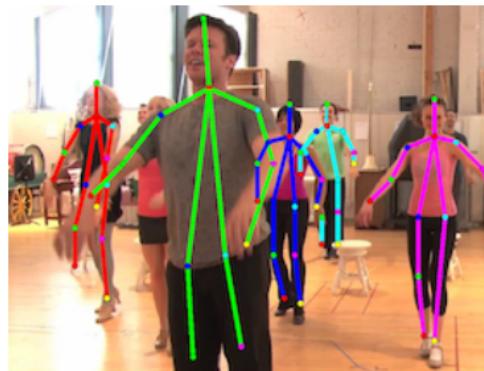
¹Een body part detector is een detector die op een 2D-beeld de verschillende lichaamsdelen van een persoon zal aanduiden of deze zal proberen te benaderen.Sigal (2014)



Figuur 2.2: DeepCut methode, Pishchulin et al. (2015)

zou zijn van een bepaalde body part class. De body part candidates worden onderverdeeld in clusters per body part class, weergegeven in Figuur 2.2b. Vervolgens wordt er voor elk paar body part candidates en elk paar body part classes een pairwise term (blauwe lijnen) gebruikt. Deze pairwise term wordt gebruikt voor het genereren van de kost of de winst voor alle mogelijke oplossingen van pose estimation, waarbij body part 1 behorende tot body part class a en body part 2 behorende tot body part class b tot dezelfde personen behoren.

DeeperCut (Insafutdinov et al. (2016)) heeft als doel om reeds bestaande technologieën te verbeteren. In een eerste fase stellen ze namelijk voor om de body part detectoren te vervangen door betere technieken die op zichzelf al uitstekende resultaten behalen op standaard pose estimation. Ze vertrekken hierbij van Residual Network (ResNet) (He et al. (2015)), en optimaliseren deze methode om aan performantie te winnen. Vervolgens introduceren ze pairwise terms tussen lichaamsdelen, die de performantie kunnen verhogen in het geval dat er meerdere mensen tegelijk in beeld zijn. Ten slotte introduceren ze een optimalisatiestrategie die ervoor moet zorgen dat de runtime verlaagd wordt en de nauwkeurigheid van de human pose estimation verbeterd wordt. Figuur 2.3 geeft een voorbeeld van de grafische weergave van het resultaat van DeeperCut.



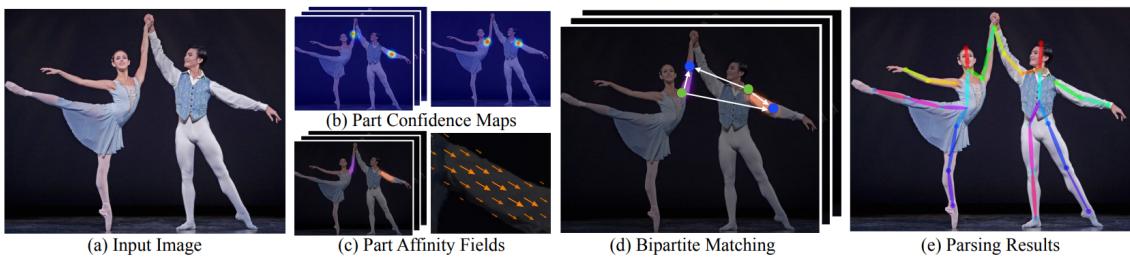
Figuur 2.3: DeeperCut output, Insafutdinov et al. (2016)

2.1.2.2 OpenPose

De state-of-the-art methode is afkomstig van OpenPose. OpenPose is een framework dat bestaat uit Convolutional pose machines (Wei et al. (2016a)) , en PAF (Cao et al. (2017)). In 2.1.2 werd al vermeld dat bottom-up pose estimation technieken goed werken voor single-person pose estimation, maar niet voor multiple-person. De reden hiervoor is dat ze niet kunnen onderscheiden welke ledematen van welke persoon zijn. Cao et al. (2017) lossen dit probleem op door gebruik te maken van PAF. Dit is een set van 2D-vectorvelden die de locatie van de verschillende ledematen in combinatie met hun oriëntatie zal coderen.

Figuur 2.4 illustreert de methode die het OpenPose framework hanteert. De input is een afbeelding in kleur (a), en de output is een verzameling van de locaties van de gewrichten van alle personen op de afbeelding (e). Deze techniek maakt, net als een heel aantal andere technieken, gebruik van een Convolutional Neural Network (CNN).

Een neuraal netwerk kunnen we vergelijken met de hersenen van de mens. De hersenen maken gebruik van neuronen om kennis op te slaan en gebruiken connecties tussen de neuronen om informatie over te dragen. De sterkte van de hersenen is de associatie tussen de verschillende neuronen. Een neuraal netwerk bestaat uit connecties tussen knopen en imiteert de associatie van de hersenen door het toevoegen van gewichten aan de connecties tussen de verschillende knopen. Net zoals de hersenen is een neuraal netwerk ook een groot netwerk. Dit wordt verwezenlijkt door te werken met verschillende lagen waarop de knopen zich bevinden. Zo hebben we aparte lagen voor de input en output, waartussen zich hidden layers bevinden. Een Deep Neural Network is een vorm van een layered neuraal netwerk dat twee of meer hidden layers bevat. Er wordt via trainingsdata een model opgesteld van wat men wil onthouden. Dit wordt dan opgeslagen in het Deep Neural Network waardoor voor een gegeven input, bij benadering, een gewenste output



Figuur 2.4: Part Affinity Fields methode, Cao et al. (2017)

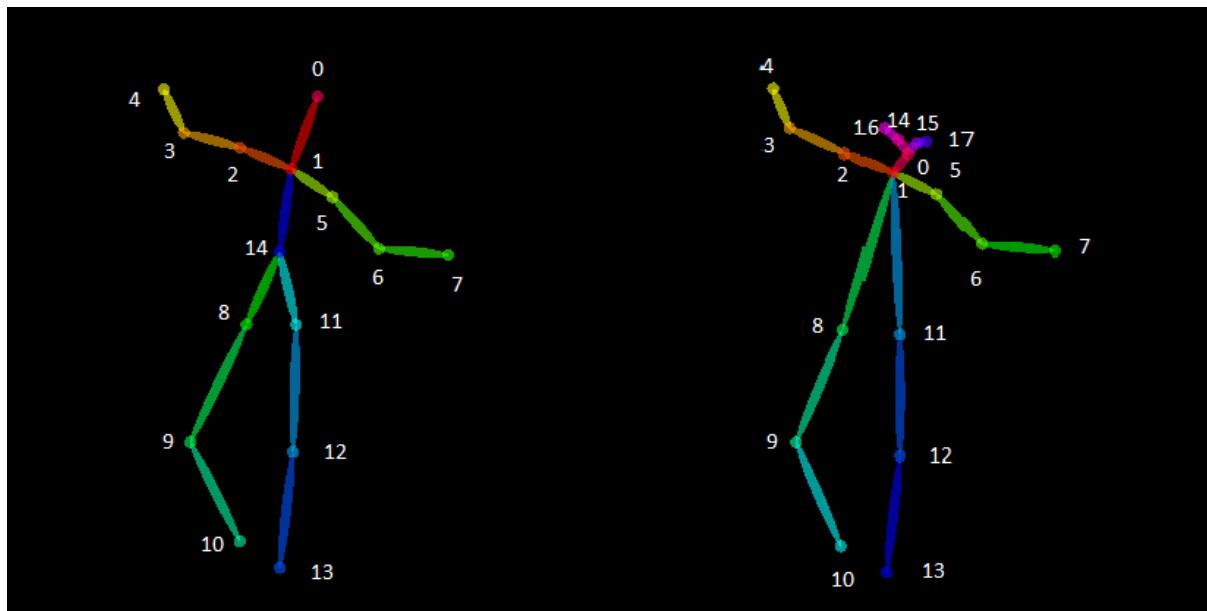
geleverd wordt. Een CNN is een Deep Neural Network gespecialiseerd in beeldherkenning. Er worden verschillende geannoteerde afbeeldingen aangeleverd aan het netwerk. Dit netwerk stelt een model op, waardoor op een gegeven afbeelding de gewenste objecten gedetecteerd worden.

De methode van Cao et al. (2017) gebeurt in verschillende stappen zoals te zien is op Fig 2.4. Eerst wordt er aan de hand van een CNN een set van 2D confidence maps van de verschillende lichaamsdelen voorspeld (b), samen met een set van 2D-vectorvelden (c), beiden volgens Wei et al. (2016a). Deze set van vectorvelden geeft de affiniteit weer om de associatie te maken tussen de verschillende lichaamsdelen van één persoon.

Uiteindelijk worden deze twee resultaten gecombineerd aan de hand van bipartite matching (d) om te kijken tot welke persoon de lichaamsdelen behoren, om daarna de volledige pose van het lichaam te benaderen. Het resultaat is een verzameling van body part locations aan de hand van een x- en y-coördinaat en een score. Het aantal punten dat als output gegeven wordt is afhankelijk van het gekozen model. Op afbeelding 2.5 is te zien dat COCO (Lin et al. (2014)) 18 keypoints heeft. MPI (Andriluka et al. (2014)) heeft er 15. Recente updates van OpenPose geven ook de mogelijkheid om de pose van handen en gezichten te benaderen (Simon et al. (2017)). Zo wordt elke hand benaderd met 21 keypoints, en heeft een gezicht er zelfs 70. De uitvoertijd voor het benaderen van de pose aan de hand van de methode van Cao et al. (2017) is onafhankelijk van het aantal personen op het frame. Dit aspect is niet van hoofdbelang voor het huidige onderzoek, aangezien ons hoofddoel is om de 3D-positie te bepalen van één persoon. Het zorgt echter wel voor meer toekomstmogelijkheden. Volgens Cao et al. (2017) haalt pose estimation aan de hand van part affinity fields snelheden van 8,8 fps voor een video met 19 mensen. Hierbij had het originele frame een resolutie van 1080x1920 px, wat herschaald werd naar 368x654 px. De gebruikte hardware voor deze test was een laptop met één NVIDIA GeForce GTX-1080 Graphics Processing Unit (GPU). Indien we handen of gezichten willen detecteren neemt de runtime wel lineair toe met het aantal mensen in de ruimte.

Op het vlak van hardware zijn er drie vereisten om te kunnen werken met het OpenPose framework:

1. NVIDIA grafische kaart met minstens 1,8GB vrij, aangezien het uitvoeren van OpenPose de



Figuur 2.5: Links: output volgens MPI Andriluka et al. (2014), Rechts: output volgens COCO Lin et al. (2014)

aanwezigheid van CUDA vereist.

2. Minstens 2GB RAM vrij.
3. Aanbeveling: NVIDIA CUDA Deep Neural Network (cuDNN) library en een Central Processing Unit (CPU) met minstens 8 cores.

2.1.2.3 Vergelijking

De methode van Cao et al. (2017) en Deepcut (Insafutdinov et al. (2016)) werden allebei uitgetest op de MPII Human Pose Dataset. De gegevens van deze dataset zijn te vinden in het onderzoek van Andriluka et al. (2014). De MPII Human Pose Dataset wordt door verschillende human pose estimation technieken gebruikt om de nauwkeurigheid te testen en te vergelijken met andere methoden. De dataset bevat ongeveer 25.000 afbeeldingen waarop zo'n 40.000 mensen te zien zijn, waarbij beredeneerd wordt waar de gewrichten van de mensen zich bevinden.

Wanneer we kijken naar de resultaten van de Single Person subset (hierbij is telkens slechts één persoon in beeld) krijgen we enkel de resultaten van Deepcut en niet die van OpenPose. Indien we kijken naar de pose estimation voor meerdere personen zien we dat OpenPose zich bij de betere pose estimation technieken mag rekenen zoals weergegeven in Tabel 2.1, en zelfs behoort tot de top drie. We zien dat de technieken van Fang et al. (2016) en Newell and Deng (2016) bepaalde gewrichten nauwkeuriger benaderen dan OpenPose, maar eigenlijk vormt OpenPose

Tabel 2.1 Vergelijking nauwkeurigheid DeeperCut en OpenPose op verschillende gewrichten. Toegepast op de Multi-Person subset van de Andriluka et al. (2014) dataset.

Methode	Hoofd	Schouder	Elleboog	Vuist	Heup	Knie	Enkel
Pishchulin et al. (2015)	78.4	72.5	60.2	51.0	57.2	52.0	45.4
Insafutdinov et al. (2016)	89.4	84.5	70.4	59.3	68.9	62.7	54.5
Cao et al. (2017)	91.2	87.6	77.7	66.8	75.4	68.9	61.7
Fang et al. (2016)	88.4	86.5	78.6	70.4	74.4	73.0	65.8
Newell and Deng (2016)	92.1	89.3	78.9	69.8	76.2	71.6	64.7

een mooie tussenmaat tussen de twee. Wanneer we de twee versies van Deepcut vergelijken, zien we dat de nieuwere versie een verbetering geeft inzake nauwkeurigheid, maar dat deze nog altijd de nauwkeurigheid van OpenPose niet kan evenaren.

Op vlak van snelheid doet DeepCut het beter dan een groot aantal andere technieken, dankzij de sterkere part detector gebaseerd op ResNet en het uitvoeren van een afbeeldingsafhankelijke pairwise score. Zoals reeds beschreven in 2.1.2.2 zorgt de nieuwe update van OpenPose ervoor dat ook de handen en het gezicht gedetecteerd kunnen worden, wat veel toekomstmogelijkheden biedt.

OpenPose haalt ook iets betere resultaten op de COCO 2016 keypoints challenge dataset (Lin et al. (2014)) dan andere technieken. DeepCut werd echter nog niet op deze dataset getest.

2.2 Bepalen van de positie in top-down view

In de tweede fase van dit onderzoek zullen we de persoon in top-down view lokaliseren. In dit hoofdstuk bespreken we twee methodes om dit te bereiken. We beginnen met kalibratie, waarna we een tweede techniek beschrijven waarbij we gebruik zullen maken van de locatie en de pose van de camera. Om de positie van een persoon ten opzichte van de camera te bepalen moet een deel van de view van deze camera overlappen met die van een tweede camera. Enkel zo krijgen we informatie over de diepte van de objecten die zich in beeld bevinden. Dit geldt voor beide technieken die we in dit hoofdstuk zullen bespreken.

2.2.1 Kalibratie

Wanneer het gaat over het positioneren van een object of persoon ten opzichte van de camera denken we al snel aan geometrische kalibratie. Heikkila and Silven (1997) definiëren geometrische kalibratie als het proces waarbij men de interne geometrie en optische karakteristieken van de ca-

mera en/of de 3D-positie en oriëntatie van het camera-frame relatief ten opzichte van een bepaald coördinaatsysteem probeert te bepalen. Men gaat dus een aantal parameters bepalen die gebruikt worden om onder andere de locatie van de camera in de omgeving te determineren. Deze parameters kunnen we onderverdelen in de intrinsieke parameters, namelijk de focale lengte en het optische centrum, en de extrinsieke parameters, namelijk de rotatie en translatie van een camera.

Voor ons onderzoek bevinden we ons echter in een speciale situatie. Zoals reeds aangegeven in hoofdstuk 1 is het mogelijk dat de camera's zich op een grote afstand van elkaar bevinden, waardoor het uitvoeren van kalibratie misschien niet de resultaten zou opleveren die we in gedachten hebben. In de meeste situaties waarbij camera's gekalibreerd worden, bevinden deze zich in elkaar's buurt. Toch bekijken we in deze paragraaf een aantal kalibratietechnieken, om te kunnen achterhalen of dit een optie is voor het huidige onderzoek. Zo bespreken we het gebruik van een dambordpatroon, het uitvoeren van self-calibration en kalibratie aan de hand van het 5-point algoritme. Voor alle technieken in verband met kalibratie gaan we ervan uit dat de cameraviews van twee camera's elkaar gedeeltelijk overlappen. Dit is een noodzakelijke aanname, omdat er bij kalibratie gezocht wordt naar punten die in beide beelden voorkomen. Hoe kleiner de hoek tussen de twee camera's, hoe makkelijker het is om een aantal punten te vinden. Indien de hoek groter wordt, zal het moeilijker worden om nog punten te vinden die voor beide camera's zichtbaar zijn, waardoor het kalibreren moeilijker en uiteindelijk onmogelijk wordt. Dit is een belangrijke beperking van kalibratie.

Ons doel is om de absolute posities van de camera's te bepalen, om zo de positie van de persoon in zijn leefruimte te definiëren. Hierbij gaat het ons dus over de extrinsieke parameters, en mogen we ervan uitgaan dat de intrinsieke parameters gekend zijn.

Enkele belangrijke begrippen die in de genoemde technieken gebruikt worden:

- De **epipolar geometry** is de intrinsieke projectieve geometrie tussen twee views. Deze is onafhankelijk van de structuur van de scène en is enkel afhankelijk van de interne parameters van de camera's, en van hun relatieve positie. Deze geometrie wordt als het ware samengevat door de fundamentaalmatrix.
- Een **fundamentaalmatrix** is een 3×3 matrix. Indien we in een driedimensionale ruimte een punt X hebben dat als x gezien wordt volgens een eerste cameraview, en als x' gezien wordt door een tweede, dan voldoen de punten aan het volgende verband:

$$x'^T F x = 0 \quad (2.1)$$

Waarbij F de fundamentaalmatrix is. De fundamentaalmatrix is onafhankelijk van de structuur van de scène, maar kan berekend worden aan de hand van puntovereenkomsten tussen twee views.

- Een **essential matrix** is een speciale vorm van de fundamentalmatrix, waarin de pixelcoördinaten genormaliseerd zijn.

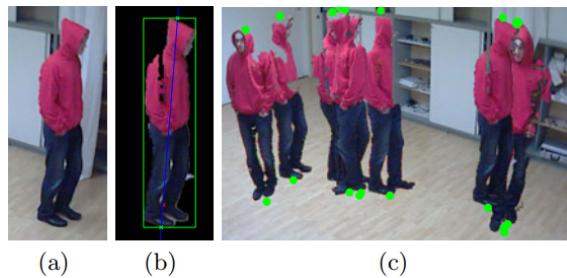
2.2.1.1 Aan de hand van gekende punten: dambordpatroon

Een traditionele manier om aan kalibratie te doen is door middel van het dambordpatroon. Bij deze techniek loopt men met een dambordpatroon door een setup, waarbij het patroon door minimum twee camera's zichtbaar moet zijn. We bewegen het dambord dan over zijn acht vrijheidsgraden, waarbij het zichtbaar blijft voor de twee camera's. Hierdoor kunnen we de relatieve positie van de camera's ten opzichte van elkaar berekenen. Dit is mogelijk omdat het dambordpatroon makkelijk te herkennen valt. Op de gemaakte beelden wordt dan een Harris Corner-detectie (Harris and Stephens (1988)) uitgevoerd die de binnenpunten van het patroon in beeld brengt. Deze punten zijn consistent in meerdere beelden, waardoor je bij de detectie van een bepaald punt in één beeld perfect kan weten met welk punt dit overeenkomt in het beeld van een andere camera dat op hetzelfde moment gemaakt is, op voorwaarde dat het dambord zich in een zone bevindt die voor beide camera's zichtbaar is. Verder wordt ook de afstand tussen twee vierkanten gebruikt. Hiervoor wordt er een metric toegevoegd, omdat de afstanden gekend zijn. Het is ook belangrijk dat één zijde van het patroon een even aantal vierkanten heeft en de andere zijde een oneven aantal. Dit is van belang om de pose van het dambordpatroon juist te detecteren. Op basis van de pose van het dambordpatroon in de twee cameraviews kan een Homography matrix² gemaakt worden waaruit de relatieve posities van de camera's berekend kunnen worden. Om de camera's te kalibreren aan de hand van een dambordpatroon moet men ter plaatse aanwezig zijn.

2.2.1.2 Self-calibration

Natuurlijk hebben we niet altijd de tijd en middelen om op voorhand de camera's te gaan kalibreren door middel van een object met een gekende structuur zoals een dambordpatroon. Om dit probleem te verhelpen bestaat er ook een andere techniek, namelijk self-calibration. Hödlmoser and Kampel (2010) bespreken een techniek die meerdere camera's in een scène kan kalibreren aan de hand van de personen die door de scène wandelen. De verschillende stappen die daarbij ondernomen worden, worden in Figuur 2.6 weergegeven. Een belangrijke kanttekening: deze techniek werd ontwikkeld voor bewakingscamera's. Bewakingscamera's hangen meestal hoger en kijken daardoor neer op de mensen die zich in de scène bevinden. Er bestonden reeds technieken om de intrinsieke parameters van de camera's te bepalen aan de hand van vanishing points (zoals de techniek van Beardsley and Murray (1992)), maar met behulp van deze techniek worden zowel de intrinsieke als de extrinsieke parameters bepaald.

²Een matrix die te vergelijken is met de fundamentalmatrix.



Figuur 2.6: Self-calibration aan de hand van voetgangers, Hödlmoser and Kampel (2010)

Hierbij wordt de persoon die door de scène loopt gedurende minstens twee frames geobserveerd, waarbij men background subtraction³ toepast en de top en bottom points verkregen worden. Deze twee punten duiden het hoogste en laagste punt van de persoon aan.

Aan het detecteren van de top en bottom points wordt extra aandacht besteed omdat de nauwkeurigheid van het algoritme hiermee valt of staat. Er wordt eerst background subtraction toegepast, om daarna de bounding box rond de persoon te berekenen (Figuur 2.6 b). Vervolgens wordt het massacentrum van de volledige bounding box berekend, samen met het massacentrum van het bovenste en onderste deel van de bounding box. Ten slotte wordt er een lijn getrokken tussen deze drie punten om de top en bottom points te vinden. Deze punten worden bepaald door de snijpunten van de getekende lijn en de bounding box (Figuur 2.6c).

De extrinsieke parameters worden gezocht door gebruik te maken van de gevonden top en bottom points. Om de absolute posities van de camera's te bepalen in wereldcoördinaten, moeten we eerst de relatieve posities van de camera's bepalen. Deze relatieve posities worden dan uitgedrukt in een relatieve translatie Δt en een relatieve rotatie ΔR .

In de begrippenlijst verklaarden we reeds dat de epipolar geometry de intrinsieke projectieve geometrie tussen twee views is, en kan voorgesteld worden door middel van de fundamentalmatrix. De volgende stap is dan ook het berekenen van deze fundamentalmatrix. Zoals eerder vermeld moet de fundamentalmatrix voldoen aan vergelijking 2.1. Volgens Hartley and Zisserman (2004) kunnen we matrix F berekenen indien we over voldoende puntenkoppels beschikken (minstens zeven). Het geval van zeven corresponderende punten vergt echter een speciale oplossingsmethode. In het geval van 8 corresponderende punten kan de matrix F opgelost worden door een set van vergelijkingen die ontstaan door het invullen van de punten in vergelijking 2.1. Dit algoritme noemt men het 8-point algorithm.

We gaan ervan uit dat de intrinsieke parameters van de camera's gekend zijn. In dat geval kunnen de genormaliseerde corresponderende punten berekend worden.

³Een techniek waarbij men de achtergrond van de bewegende objecten onderscheidt.

De verschillende punten worden genormaliseerd door middel van een translatie en schaling van elke afbeelding om het resultaat van de lineaire vergelijkingen te verbeteren. Het uitvoeren van het 8-point algorithm op deze koppels heeft de essential matrix als resultaat. De twee grootheden die we zoeken, de relatieve translatie en rotatie, kunnen rechtstreeks uit de essential matrix berekend worden (zie formule 2.2) . Hierbij is E de essential matrix, Δt de relatieve translatie en ΔR de relatieve rotatie.

$$E = [\Delta t]_x \Delta R = \Delta R [\Delta R^T \Delta t]_x \quad (2.2)$$

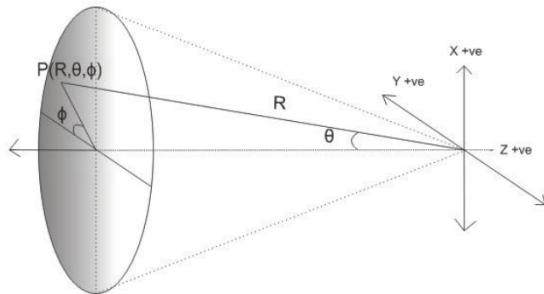
Deze formule leidt echter tot vier verschillende oplossingen. Er moet hier enkel naar de oplossing gekeken worden die een positieve diepte heeft voor alle geprojecteerde punten.

Alle vorige stappen werden uitgevoerd om de relatieve positie van de camera's ten opzichte van elkaar te bepalen. Om de absolute positie van de camera's te bepalen, plaatsen we één camera in de oorsprong van het wereldcoördinatenstelsel. Dankzij de relatieve translatie en rotatie, in combinatie met de top en bottom points van de persoon en zijn hoogte, kan de absolute oriëntatie voor elke camera bepaald worden. De snelheid van dit algoritme voor respectievelijk twee, drie, vier en vijf camera's is: 1.5, 2.7, 4.3 en 5.6 seconden. Dit komt ongeveer overeen met een toename van de rekentijd van 1.4 seconden per camera.

Hödlmoser and Kampel (2010) verklaren niet welke acht punten gebruikt worden voor deze techniek of wat de maximale afstand mag zijn tussen de twee camera's. Dit is echter van groot belang aangezien de acht punten die gebruikt worden voor de kalibratie zichtbaar moeten zijn in beide views, en onze camera's zich op een grote afstand van elkaar kunnen bevinden. We weten wel dat de top en bottom points altijd zichtbaar moeten zijn voor de camera's. De persoon moet dus helemaal in beeld zijn en er mag geen occlusie plaatsvinden. Voor de top en bottom points maakt de afstand niet uit, aangezien er van de techniek uit geen maximum afstanden gegeven zijn tussen de camera's en de bounding box altijd getekend wordt, onafhankelijk van de plaats van de camera.

2.2.1.3 5-Point Algorithm

Het 5-Point Algorithm (Nisér (2004)) geeft een oplossing voor het 5-Point Relative Pose Problem. Het gegeven probleem is het vinden van de mogelijke relatieve posities van twee camera's ten opzichte van elkaar indien de intrinsieke parameters zoals de focal length van de camera gekend zijn, en dit aan de hand van vijf koppels van punten. In de vorige paragraaf over self-calibration werd reeds gesproken over een 8-point algorithm. Het 5-point algorithm is dan ook soortgelijk. De input van het algoritme bestaat uit vijf paren van punten. Eén paar wordt gevormd door de selectie van een punt op de ene afbeelding en het corresponderende punt op de tweede afbeelding.



Figuur 2.7: θ en Φ voor positiebepaling van persoon, Hombali et al. (2011)

Het is hierbij niet gekend over welk punt het gaat in de ruimte en wat de viewpoints zijn van de twee camera's. Het doel van het algoritme is het bepalen van de coëfficiënten van een 10^{de} graads polynoom om uiteindelijk zijn nulpunten te bepalen. Het resultaat is dan een benadering van de essential matrix E, de rotation matrix R en de translation vector t tussen de twee afbeeldingen. De werkwijze van dit algoritme is zeer complex en wordt beschreven door Nisér (2004).

Kalibratie is een nauwkeurige methode om afstanden tussen punten te kunnen bepalen. Er is echter ook een nadeel aan verbonden. Indien we op een bepaalde zone geen kalibratie kunnen uitvoeren doordat deze maar zichtbaar is voor één camera, beschikken we over te weinig informatie om een plaatsbeschrijving te geven.

2.2.2 Angle to pixel ratio

In het onderzoek van Hombali et al. (2011) wordt een manier besproken om diepte-informatie te halen uit één enkele afbeelding. Het doel hierbij is het bepalen van de diepte van een object ten opzichte van de camera, en dit op onbekend terrein en zonder gegevens over de hoogte van de camera. Ook de positie van het object ten opzichte van de camera wordt bepaald. Deze techniek maakt gebruik van een lijn tussen de camera en het object, die gekenmerkt wordt door twee variabelen: θ en Φ , weergegeven in Figuur 2.7. Hierbij is θ de hoek tussen de loodlijn van het object naar het centrum van de camera en de positieve x-as. De loodlijn komt overeen met de lijn tussen de pixel die de positie van het object weergeeft en het centrum van de afbeelding. Φ is de hoek tussen diezelfde pixel en de as waarop de camera ligt. Het is dus eigenlijk een maat voor de hoogte van de gegeven pixel ten opzichte van de camera. Vervolgens wordt er een constante berekend: de angle to pixel ratio (AP). Dit is de verhouding tussen de hoek van de view in verticale richting en het aantal pixels in verticale richting. Aangezien dit twee constanten zijn, is de AP dus ook constant. De hoek Φ wordt dan berekend door het product te nemen van de AP-constante en de y-waarde van de pixel die overeenkomt met het object.

2.2.3 Vergelijking kalibratie en angle to pixel ratio

We hebben verschillende vormen van kalibratie besproken, elk met een verschillende werkwijze. Er bestaat geen twijfel over dat kalibratie aan de hand van een dambordpatroon nauwkeurig zou zijn. De vraag is echter of deze methode geschikt is om te gebruiken indien de camera's zich ver van elkaar bevinden. Het dambordpatroon moet immers, zoals eerder vermeld, zichtbaar zijn voor beide camera's gedurende het uitvoeren van de acht vrijheidsgraden. De self-calibration methode van Hödlmoser and Kampel (2010) kan handig zijn in ons onderzoek, zeker aangezien deze kalibreert aan de hand van mensen en ons onderzoek juist gebaseerd is op mensen. De auteurs vermelden echter niet welke punten er gebruikt worden voor het genormaliseerde 8-point algorithm. Ook dit kan een rol spelen indien de camera's verder van elkaar verwijderd worden. Hier geldt immers eenzelfde voorwaarde als bij het dambordpatroon: de punten moeten zichtbaar zijn voor beide camera's.

Het 5-point algorithm is een handig algoritme, aangezien de punten manueel aangeklikt kunnen worden en we dus zelf punten kunnen kiezen die zichtbaar zijn voor beide camera's. De vraag is echter of kalibratie de oplossing is voor dit onderzoek, aangezien de afstand tussen de camera's groot kan worden. Ook geeft kalibratie weinig tot geen informatie over zones die slechts zichtbaar zijn voor één camera. In deze situatie zal de angle to pixel ratio iets meer informatie geven, aangezien deze methode een lijn genereert waarop de persoon zich bevindt. De vraag is echter ook hoe belangrijk nauwkeurigheid in dit onderzoek zal zijn. Het doel is namelijk om een 3D pose estimation te vormen van de persoon in beeld, niet om een gedetailleerde beschrijving te geven van zijn positie in de ruimte.

2.2.4 Probabilistic Occupancy Map

Tijdens de tweede fase van ons onderzoek ontdekten we een techniek genaamd Probabilistic Occupancy Map (POM) (Fleuret et al. (2008)). Deze techniek geeft elk punt in een bepaalde zone een score, die aangeeft wat de kans is dat een persoon zich op deze positie bevindt. Deze locaties worden vervolgens gebruikt om de verschillende aanwezige personen te tracken. De input van het algoritme bestaat uit binaire afbeeldingen die bekomen worden door het uitvoeren van background subtraction op de beelden van alle camera's. Hoewel het resultaat van deze techniek lijkt op wat we ook met het huidige onderzoek willen bekomen, zijn er toch een aantal verschillen op te merken. Zo wordt de detectie van personen op de camerabeelden uitgevoerd met behulp van background subtraction. Het doel van het huidige onderzoek is echter om een 2D-pose estimator te gebruiken om zo meer informatie te krijgen over de pose van de persoon dan background subtraction. Op deze manier hebben we meer informatie om de 3D-pose te bepalen. Fleuret et al. (2008) vermelden in hun onderzoek niet wat de maximumhoek is die twee camera's ten opzichte van elkaar

kunnen maken, waarvoor het algoritme nog steeds zou werken. We kunnen dit ook niet afleiden uit hun werkwijze, die zeer wiskundig is. De auteurs vermelden wel dat het algoritme werkt indien de camera's zich in de hoeken van een kamer bevinden. In het huidige onderzoek is het doel om de 3D-pose te bepalen in een leefruimte. daardoor is het mogelijk dat de hoek tussen de camera's groter is dan 90 graden. Aangezien POM toch wat in de lijn ligt van ons onderzoek hebben we de testimplementatie uitgevoerd. De structuur van de configuratiefile bleek echter moeilijk te begrijpen, waardoor we deze niet op eigen data konden uitvoeren. Dit probleem, in combinatie met de verschillen tussen het huidige onderzoek en POM én het feit dat we deze techniek pas ontdekten toen we reeds bezig waren met het implementeren van onze eigen techniek, heeft tot de beslissing geleid om verder te gaan met onze eigen implementatie en pom niet te gebruiken.

2.2.5 Test-datasets

Indien we na implementatie onze techniek willen uittesten, moeten de top-down posities die wij berekend hebben kunnen vergeleken worden met de werkelijke positie. We zullen onze techniek op een (bestaande) dataset moeten valideren. Daarom zoeken we een dataset waarin de positie bepaald wordt aan de hand van de dieptegegevens op beelden. Er zijn echter een aantal bijkomende eisen waaraan de dataset moet voldoen:

- De beelden moeten gefilmd zijn met een Red Green Blue Depth (RGBD) camera, zodat we informatie hebben over de diepte om de positie van de persoon te kunnen bepalen
- Er moet met meerdere camera's gefilmd worden, waarbij er overlap is tussen de verschillende beelden
- Er mag telkens maar één persoon tegelijk in beeld zijn
- De persoon in beeld moet zich verplaatsen
- Het moeten statische camera's zijn
- De beelden moeten gefilmd zijn in een leefruimte
- Bij voorkeur moet er één camera de persoon op ooghoogte filmen

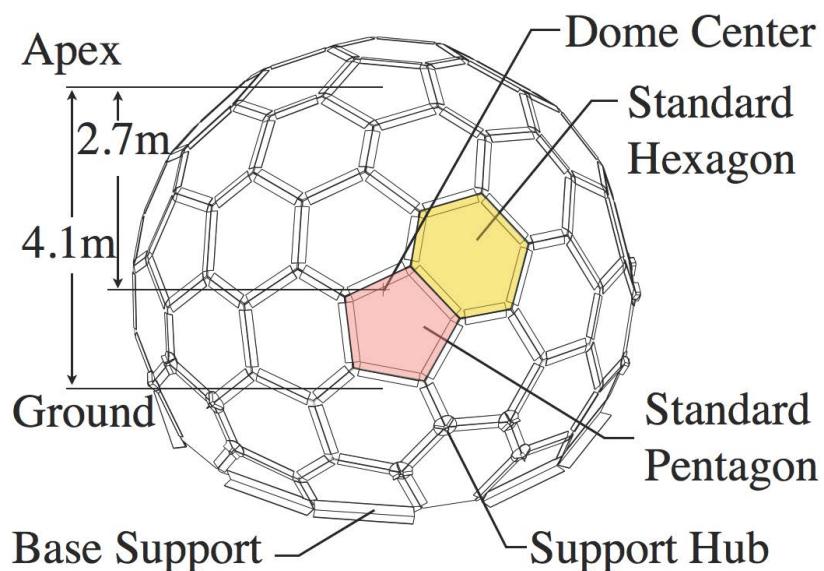
Dit zijn echter erg strenge criteria, waardoor het aantal datasets dat hieraan voldoet zeer beperkt is. Zelfs wanneer we het laatste criterium laten vallen, is het aantal mogelijke datasets nog beperkt. Er zijn een groot aantal datasets beschikbaar voor human activity analysis zoals de datasets van Shahroudy et al. (2016) en Liu et al. (2017). Deze datasets zijn vaak gemaakt met meerdere camera's met gedeeltelijk overlappende views, maar de personen op de beelden staan zo goed als stil. Daarnaast zijn er ook de reeds besproken datasets van Andriluka et al. (2014) en Lin et al.

(2014). Deze zijn bedoeld voor object- of persoonsdetectie en voor pose estimation. De beperking bij deze datasets is dat er maar met één camera gefilmd wordt, of er maar één foto beschikbaar is van een bepaalde scène in plaats van een sequentie. Dit maakt deze datasets onbruikbaar voor het uittesten van onze techniek.

Een dataset die voor het huidige onderzoek wel nuttig zou kunnen zijn, is de CMU Panoptic Dataset, (Joo et al., 2015). De beelden die deze dataset bevat zijn wel niet gemaakt in een leefruimte. Deze dataset is verkregen via de Panoptic Studio, een systeem dat oorspronkelijk ontwikkeld werd voor een masterproef waarin de sociale interactie bestudeerd werd aan de hand van een samenstelling van verschillende viewpoints. Het systeem bestaat uit 480 gesynchroniseerde VGA-camera's met een resolutie van 640 x 480 px, 31 gesynchroniseerde HD-camera's met een resolutie van 1920 x 1080 px, 10 Kinect II Sensors met een resolutie van 1920 x 1080 px RGB en 512 x 424 px diepte. De VGA- en HD-camera's zijn gesynchroniseerd aan de hand van een hardware klok en hun timing is gelijk. De Kinect-sensoren lopen echter op een andere timing. Figuur 2.8 geeft een beeld van hoe de studio is opgebouwd. Zo zien we dat de studio een bolvormige figuur is, volledig opgebouwd aan de hand van zeshoeken. Op deze manier kunnen er beelden gemaakt worden van de personen of voorwerpen die zich in de studio bevinden en dit vanuit verschillende perspectieven. De verschillende types camera's worden weergegeven in Figuur 2.9. Op deze figuur zien we dat in het centrum van de meeste zeshoeken een HD-camera geplaatst is. Ook zien we dat er meerdere VGA-camera's in een zeshoek geplaatst zijn (wat ook het groot aantal VGA camera's verklaart), en er maar een aantal Kinect-camera's bevestigd zijn. De dataset bevat verschillende beeldfragmenten. Zo zijn er fragmenten waarbij er meerdere personen in beeld zijn, maar er zijn ook fragmenten waarop slechts één persoon te zien is. Ook worden er verschillende handelingen afgebeeld. Er zijn fragmenten waarbij we zien dat de personen een spel spelen, dansen, een muziekinstrument bespelen en nog een aantal andere videosequenties. Door het vele aantal camera's is het echter moeilijk zones te vinden die maar voor één camera zichtbaar zijn.

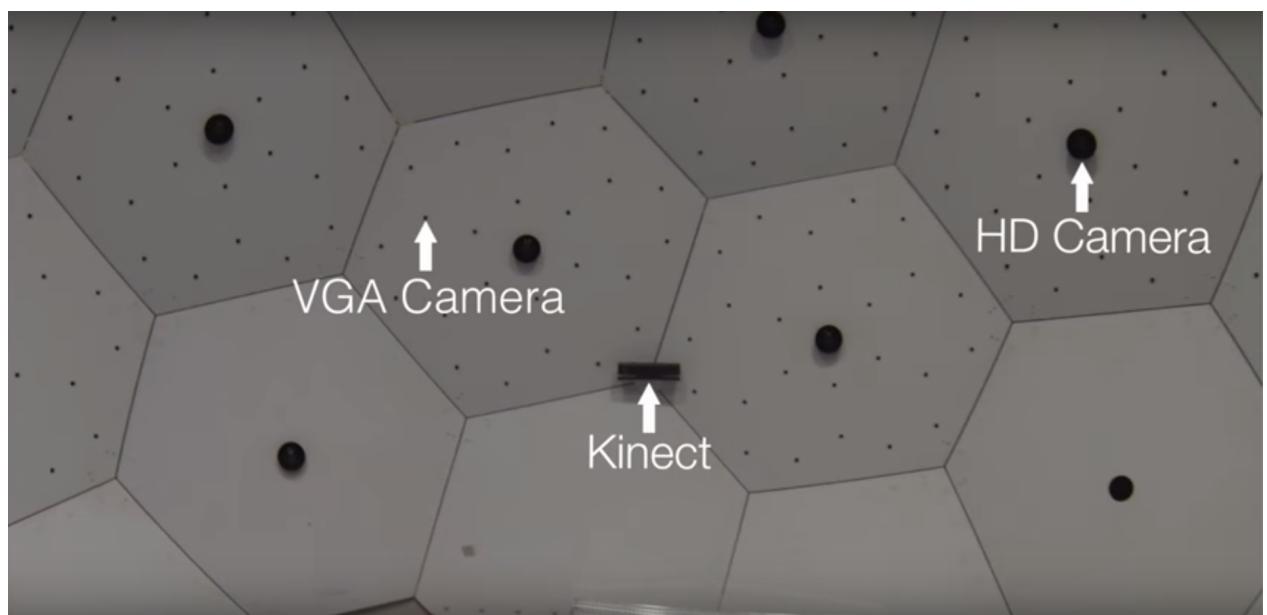
2.3 3D-pose

Het vormen van een 3D pose estimation kunnen we op twee verschillende manieren aanpakken. Ten eerste kunnen we een 3D-pose benaderen op basis van meerdere views. Ten tweede kunnen we dit ook doen aan de hand van één enkele view.



Figuur 2.8: Opbouw Panoptic Studio

Joo et al. (2015)



Figuur 2.9: Camera types Panoptic Studio

Joo et al. (2015)

2.3.1 Aan de hand van meerdere views

Er bestaan verschillende methodes om de 3D-pose van een persoon aan de hand van meerdere views te benaderen. Zo vermeldden we reeds het onderzoek van Ershadi-Nasab et al. (2017), waarin de 3D-pose van meerdere mensen benaderd wordt aan de hand van beelden uit meerdere views. Deze past kalibratie toe. Ook in de onderzoeken van Pavlakos et al. (2017), Amin et al. (2013) en Hofmann and Gavrila (2012) wordt deze methode gebruikt. Dit is zeker niet onlogisch, aangezien we een verband moeten kennen tussen de views van een camerakoppel om informatie van beide camera's te combineren tot een geheel. In het huidige onderzoek is de situatie echter anders, aangezien de afstanden tussen de verschillende camera's groot kan zijn.

Het doel van het onderzoek van Pavlakos et al. (2017) is het bepalen van de 3D-pose van een mens door een CNN toe te passen, dat op elke view confidence heatmaps genereert voor elk gewicht. Bovendien legt deze methode een aantal constraints vast, door gebruik te maken van de 3D-geometrie van de camera-setup en de natuurlijke limieten van het menselijk lichaam. Er is slechts een deel van de implementatiecode publiek beschikbaar. Amin et al. (2013) stellen een multi-view pictorial structures model (Fischler and Elschlager (1973)) voor. Deze auteurs beginnen met het benaderen van de 2D-pose van de persoon in beeld, en dit voor elke view. Vervolgens maken ze de stap naar 3D door middel van triangulatie. De nauwkeurigheid van beide technieken wordt in het onderzoek van Ershadi-Nasab et al. (2017) vergeleken met andere technieken, waarbij we zien dat de resultaten van Amin et al. (2013) door een aantal technieken overtroffen worden en de resultaten van Pavlakos et al. (2017) iets onderdoen voor die van Ershadi-Nasab et al. (2017). Hofmann and Gavrila (2012) presenteren ook een techniek die de 3D-pose wil bepalen aan de hand van meerdere views. Het nadeel van deze techniek is dat hij enkel de pose benadert van het bovenlichaam.

2.3.1.1 Multiple human 3D pose estimation from multiview images

Multiple human 3D pose estimation from multiview images, beschreven door Ershadi-Nasab et al. (2017), bepaalt de 3D-pose van meerdere mensen in een ruimte. In ons onderzoek gaan we slechts uit van één persoon in de ruimte, wat het zoeken naar de 3D-pose iets vergemakkelijkt. Deze techniek houdt rekening met de mogelijkheden van de menselijke gewrichten, en gebruikt de output van een 2D deep part detector. De onderzoekers stellen een conditional random field (CRF)⁴ voor. Deze techniek wordt uitgevoerd in drie verschillende stappen:

1. **shared 3D search space bepalen.** De eerste stap is het bepalen van de zone waarin de personen aanwezig zijn. Deze zone wordt de shared 3D search space genoemd.

⁴Een probabilistische methode voor gestructureerde voorspellingen. Sutton and McCallum (2012)

Op de afbeeldingen van de verschillende views wordt de DeeperCut 2D part detectie uitgevoerd. Het resultaat is zoals reeds besproken in sectie 2.1.2.1 een aantal score maps van de verschillende gewrichten. Deze score maps bestaan uit een array met grootte $W \times H \times 15$, waarbij W de breedte is van de afbeelding, H de hoogte en 15 het aantal gewrichten. Aan de hand van de score maps en de camerakalibratiematrix gaat men bepalen welke punten het meest waarschijnlijk tot de persoon behoren, en voegt men deze toe aan de 3D search space.

2. **Clusteren van shared space.** Bij het clusteren gaan we een verzameling van gegevens, in dit geval punten, groeperen zodat punten in eenzelfde verzameling iets gemeen hebben. In dit geval gaat het om punten die tot dezelfde persoon behoren. Het resultaat van de vorige stap is een verzameling van punten die toebehoren aan verschillende mensen. Nu moet er onderzocht worden welke punten tot welke persoon behoren. De search space wordt dus onderverdeeld in verschillende afzonderlijke delen. Zo kan de overgang van een 2D-pose naar een 3D-pose aangepakt worden per persoon. Het Gaussian mixture model (Rasmussen (2000)) wordt uitgevoerd voor een verschillend aantal personen tussen één en tien, omdat we op voorhand niet weten hoeveel mensen er exact in de scène aanwezig zijn. Met deze methode wordt het aantal mensen in de scène geschat en worden de punten aan de clusters toegekend. Het model met de beste waarde voor het Akaike's information criterion (Akaike (1974)) wordt gekozen.
3. **3D pose estimation.** Uiteindelijk worden de search space per persoon, de score maps en de camera calibration matrices gebruikt om de unary⁵ en pairwise terms van elke persoon te berekenen. Zoals reeds vermeld wordt de pose benaderd door gebruik te maken van een CRF, die gebaseerd is op het voorstellen van de persoon door een boomvoorstelling. Hiervoor worden de relaties tussen de gewrichten die naast elkaar liggen genoteerd in de vorm van Euclidische constraints. Er wordt hierbij onder andere rekening gehouden met de vrijheidsgraden van de verschillende gewrichten. Voor de constraints van de gewrichten die niet naast elkaar liggen, wordt de output van DeeperCut gebruikt.

Volgens dit onderzoek heeft men minimum drie camera's nodig om een 3D-pose te kunnen benaderen. Bij het gebruik van twee camera's kunnen er namelijk al problemen optreden indien de persoon door occlusie voor één van de camera's maar gedeeltelijk zichtbaar is. In verband met de maximale afstand tussen de verschillende camera's en de hoogte van de camera's geven de auteurs geen informatie. Wel worden er een aantal resultaten getoond waarbij de camera's op verschillende hoogtes hangen, wat positief is voor onze toepassing. In verband met de afstand tussen de verschillende camera's lijkt het alsof ze op een relatief kleine afstand van elkaar gepositioneerd

⁵Een schatting over de locatie van de lichaamsdelen in de afbeelding.

Tabel 2.2 Nauwkeurigheid van de verschillende technieken op de Ionescu et al. (2014) dataset

Techniek	Discussiëren	Eten	Groeten	Fotograferen	Wandelen	Hond uitlaten	Zitten
Tekin et al. (2016)	129.06	91.43	121.68	162.17	65.75	130.53	-
Li and Chan (2014)	183.09	132.50	162.27	206.45	97.07	177.84	-
Pavlakos et al. (2016)	-	-	-	-	59.12	-	76.99
Mehta et al. (2016)	68.58	59.56	67.34	82.40	55.24	76.50	99.98
Tekin et al. (2015)	102.41	88.83	125.28	182.73	55.07	126.29	138.89
Yasin et al. (2015)	72.5	108.5	110.2	142.5	92.1	165.7	119.0
Zhou et al. (2015)	109.31	87.05	103.16	143.32	79.39	114.23	124.52
Mehta et al. (2017)	78.1	63.4	72.5	93.8	55.8	82.0	106.6
Chen and Ramanan (2016)	66.60	74.74	79.09	93.26	55.74	85.86	90.74

zijn. Een benodigde input van het systeem zijn de camerakalibratiematrixes. Voor het kalibreren verwijzen we naar hoofdstuk 2.2.1. Het systeem werd geëvalueerd op verschillende multiview en multiple human datasets. Er werd echter slechts op één dataset getest die zich in een leefruimte afspeelt, namelijk de MPII Cooking dataset (Schiele (2012)).

2.3.2 Aan de hand van single view

Het benaderen van een 3D-pose aan de hand van één enkele afbeelding is een uitdaging. Er zijn heel wat factoren die dit bemoeilijken, zoals occlusie, complexe achtergronden en het verlies van 3D-informatie bij de overgang naar 2D.

In wat volgt bespreken we een aantal technieken en vergelijken we hun nauwkeurigheden. Een vaak gebruikte dataset voor het bepalen van de nauwkeurigheid van een gevormde 3D-pose is de dataset van Ionescu et al. (2014). Deze dataset bevat 3,6 miljoen 3D-poses, gemaakt door 11 acteurs in 17 scenario's. Hierbij wordt de fout tussen de werkelijke en de gevonden 3D-positie in millimeter weergegeven.

Tekin et al. (2016) beschrijven een techniek die de constraints voor het menselijk lichaam zal vastleggen aan de hand van een auto-encoder. Vervolgens zal hun CNN het inputbeeld mappen met deze constraints om vervolgens de 3D-pose te bekomen.

Ook Li and Chan (2014) maken gebruik van een CNN. Ze trainen het netwerk op twee manieren. Een eerste mogelijkheid is om per gewicht de positie ten opzichte van het rootgewicht te voorspellen en op die manier pose regression en body part detectors te trainen. Een tweede mogelijkheid is om het resultaat van bestaande part detectors om een pose te initialiseren en hierop regressie toe te passen.

Mehta et al. (2016) introduceren een nieuwe dataset om een CNN te trainen. Om de 3D-pose van een persoon te bepalen, halen ze aan de hand van 2D-detecties de bounding box uit de input-afbeelding. Vervolgens voeren ze met behulp van een CNN pose regression uit, om daarna de 3D-pose weer te geven op de originele afbeelding. De laatste aanpassingen gebeurden in 2017, wat het ook niet onlogisch maakt dat er geen code publiek beschikbaar is.

In het onderzoek van Wang et al. (2014) wordt de pose benaderd door een aantal stappen te volgen. Eerst wordt op de afbeelding de 2D-pose benaderd en wordt er een 3D-pose geïnitialiseerd. Vervolgens worden de cameraparameters benaderd aan de hand van de 2D- en 3D-pose, waarna de 3D-pose wordt aangepast op basis van deze informatie. Deze laatste twee stappen blijft men herhalen tot de fout tussen de projectie van de 3D-pose volgens de cameraparameters en de 2D-pose zo klein mogelijk is. Hierbij worden er bovendien eisen opgelegd in verband met de lengte van de ledematen.

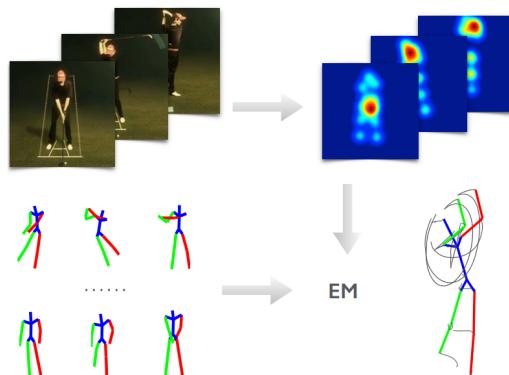
Tekin et al. (2015) bespreken in hun onderzoek een techniek die door middel van motion-informatie van opeenvolgende frames van een videosequentie de 3D-pose voorspelt. De persoon wordt gedurende een aantal opeenvolgende frames gedetecteerd. Aan de hand van een CNN wordt het image window verplaatst, zodat de persoon centraal in beeld blijft. Vervolgens wordt er een volume gevormd over de verschillende image windows in de tijd, om hier 3D HOG features uit te halen. Vervolgens wordt de 3D-pose gevormd door middel van regressie.

Yasin et al. (2015) gebruiken twee aparte datasets: één met geannoteerde 2D-poses en één dataset met nauwkeurige 3D motion data die enkel 3D-poses bevat en opgenomen is in een labo. Ze stellen een dual-source aanpak voor waarbij uit de verschillende 3D-poses de bijhorende 2D-pose gevonden wordt door middel van projectie. Deze 3D-poses worden dan vergeleken met de gevonden 2D-pose op de input-afbeelding, en de beste 3D-pose wordt gekozen aan de hand van de kleinste projectiefout.

Pavlakos et al. (2016) bekijken het benaderen van een 3D-pose als het lokaliseren van keypoints in een discretized 3D-ruimte. Ze interpreteren de 3D-ruimte dus bekijken als een aantal 2D-vlakken die op elkaar aansluiten. Op die manier trainen ze een CNN om, met een afbeelding als input, in dit volume per voxel⁶ te voorspellen of een gewicht hierin zal voorkomen of niet.

In Tabel 2.2 vindt u de nauwkeurigheden van de reeds genoemde technieken, getest op de dataset van Ionescu et al. (2014). De techniek van (Mehta et al., 2016) haalt bij de meeste handelingen de beste score, maar hiervan is er momenteel geen code publiek beschikbaar. Twee andere technieken die opvallend goed scoren op vlak van nauwkeurigheid zijn die van Mehta et al. (2017) en Chen and Ramanan (2016). Het voordeel van de methode van Mehta et al. (2017) is, buiten zijn nauwkeurigheid, dat het een techniek is die de 3D-pose in realtime zal benaderen.

⁶Een punt in een driedimensionale ruimte.



Figuur 2.10: Sparseness Meets Deepness overview

Zhou et al. (2015)

Dit kan in het huidige onderzoek van grote waarde zijn. In sectie 2.3.3 bespreken we de techniek van Zhou et al. (2015), omdat deze over het algemeen goede resultaten haalt en publiek beschikbaar is. We bespreken de technieken van Chen and Ramanan (2016) en Mehta et al. (2017) in secties 2.3.4 en 2.3.5.

2.3.3 Sparseness Meets Deepness

Zhou et al. (2015) combineren de 2D-locaties van de verschillende gewrichten met een geschat model van de 3D-pose aan de hand van een Expectation-Maximization (Moon (1996)), om zo de 3D-pose te benaderen. Hierbij kunnen de 2D-locaties van de verschillende gewrichten aangeleverd worden als input, of door de techniek zelf afgeleid worden. In het tweede geval wordt een CNN getraind om een heatmap te vormen van de 2D-lichaamslocaties. Het systeem zal de lichaamsdelen in 2D voorspellen op basis van de afbeelding, en de 3D-pose reconstrueren op basis van een model. Om fouten van de 2D part detector te verminderen past men temporal smoothness toe op de 3D-pose en de viewpoint parameters. Het 3D-model is aangeleerd op basis van training poses.

2.3.4 2D Pose Estimation + Matching

Een andere methode voor het vinden van de 3D-pose wordt voorgesteld in het onderzoek van Chen and Ramanan (2016). Hun werkwijze is geformuleerd volgens vergelijking 2.3.

$$p(X, x, I) = p(X|x, I) \cdot p(x|I) \cdot p(I) \quad (2.3)$$

Waarbij X de 3D-pose is, x de 2D-pose en I de afbeelding. Na aanname dat de 3D-pose niet afhankelijk is van de afmetingen van de afbeelding wordt dit vereenvoudigd naar:

$$p(X, x, I) = p(X|x).p(x|I).p(I) \quad (2.4)$$

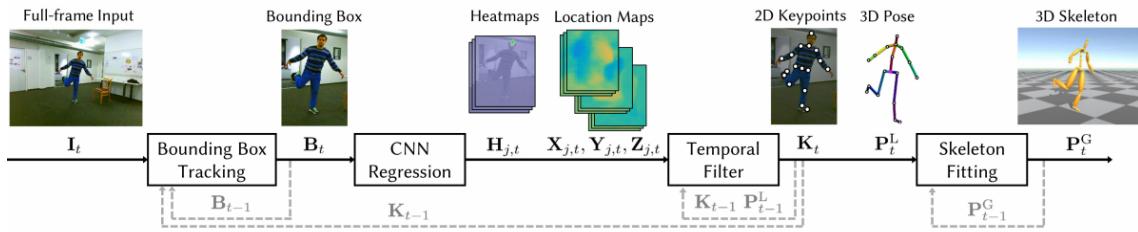
Uit vergelijking 2.4 kunnen we de stappen van het onderzoek afleiden:

1. De auteurs leiden de 2D-pose af uit één enkele RGB-afbeelding door middel van een convolutional neural network (CNN). Dit komt overeen met de ' $p(x|I)$ '-term van de vergelijking. Het resultaat is (zoals we reeds gezien hebben) een verzameling van 2D heatmaps van de gewrichten. Hierbij wordt er gebruik gemaakt van convolutional pose machines (CPM) Wei et al. (2016b).
2. Na het verkrijgen van de 2D-pose wordt de ' $p(X|x)$ '-term aangepakt. Dit is de kans dat een bepaalde 3D-pose toebehoort aan de verkregen 2D-pose. We beschikken over een library van 3D-poses, waarvan we de verschillende 2D-poses kunnen bekomen door het gebruik van een camera projection matrix. De gevonden 2D-pose uit de vorige stap wordt dan gematcht met de verschillende 2D-poses van de 3D-poses in de library. Door het toepassen van het nearest-neighbor model, wordt dan de 3D-pose gekozen.

Dit alles gebeurt in een tijd die korter is dan 200ms, wat overeenkomt met een snelheid van 5 frames per seconde. Van deze 200ms worden er 160 gebruikt voor het benaderen van de 2D-pose en 26ms voor het matchen van de verkregen 2D-pose met een library van 200 000 poses. Het feit dat het systeem opgedeeld is in twee fasen geeft het voordeel dat er gebruik gemaakt wordt van twee verschillende types trainingssets. Eén voor het trainen van de 2D pose estimation en een andere voor de 3D reasoning module. Het systeem is getest op de Human3.6M (Ionescu et al. (2014)) en de LSP dataset (Johnson and Everingham (2010)), waarin ook voorbeelden zijn met oclusie en zittende personen. Dat er met behulp van deze techniek reeds een 3D-pose gevormd kan worden op basis van slechts één view is een mooi resultaat en open heel wat deuren. Dit zou betekenen dat we in staat zijn om de 3D-pose te benaderen in het geval er één tot drie camera's aanwezig zijn.

2.3.5 VNect

VNect (Mehta et al. (2017)) is een veelbelovende techniek, die gebaseerd is op Mehta et al. (2016). Het is een techniek die het mogelijk maakt om realtime de 3D-pose van een persoon te benaderen. Figuur 2.11 wordt de gevuldte werkwijze geïllustreerd. De input van het systeem is een continue opeenvolging van Red Green Blue (RGB) afbeeldingen. Het systeem bestaat uit een CNN, die door middel van regressie de 2D- en 3D-gewrichtslocaties te bepaalt.



Figuur 2.11: VNect overview

Mehta et al. (2017)

De 2D-gewichtslocaties worden weergegeven in de heatmaps, terwijl de 3D-gewichtslocaties, relatief ten opzichte van de root, worden weergegeven in location-maps. Deze waarden worden door deze techniek ook als output gegenereerd. Het CNN is getraind op geannoteerde data van de Ionescu et al. (2014) dataset en een nieuwe dataset, voorgesteld door Mehta et al. (2016). Uit de heatmaps kunnen de 2D keypoints afgeleid worden. De 2D-voorspellingen worden gebruikt om de 3D-coördinaten van elk gewricht te bepalen uit de location maps. Vervolgens wordt er een energy-functie opgesteld in functie van de hoeken die de verschillende gewrichten maken en de locatie van het rootgewicht in de cameraruimte. De 2D- en 3D-voorspellingen worden dan gecombineerd door het minimum van deze functie. De auteurs noemen dit skeleton fitting. Het systeem is uitgerust met een default skelet, wat voor de meeste mensen goed werkt. Men kan de nauwkeurigheid verhogen door een skelet te initialiseren, door een CNN een aantal voorspellingen te laten doen over de relatieve lengtes van de ledematen van de persoon in beeld, en dit toe te passen op het skelet. Hiervoor hebben we de hoogte (afstand tussen hoofd en tenen) van de persoon in beeld nodig.

De nauwkeurigheid van deze techniek wordt getoond in Tabel 2.2, waaruit blijkt dat VNect één van de nauwkeurigere technieken is. Toch faalt ook deze techniek in een aantal gevallen. Zo heeft VNect moeilijkheden wanneer het gezicht geocludeerd is, als er zeer snel bewogen wordt, als er meerdere personen tegelijk in beeld zijn of wanneer de 3D-pose te veel afwijkt van de training set. De berekeningen van het CNN duren 18ms, de skeleton fitting 7 tot 10ms en de preprocessing en filtering duren 5ms, wat een totale duur geeft van ongeveer 33ms. Deze snelheden worden gehaald met een 6-core Xeon CPU met een kloksnelheid van 3.8GHz en een Titan X GPU. Dit komt overeen met een snelheid van ongeveer 30Hz.

Afgaande op de literatuur omtrent deze techniek, lijkt hij zeker bruikbaar voor dit onderzoek. Het is een techniek die realtime werkt, en zich tot één van de meest nauwkeurige technieken mag rekenen. Verder is het ook zeer interessant dat de 2D-gewichtslocaties ook als output gegenereerd worden, zonder al te veel overhead. Het is dus een mogelijkheid om geen afzonderlijke 2D pose estimation techniek te gebruiken, maar enkel de output van VNect. De nauwkeurigheid van de 2D pose estimation wordt echter niet weergegeven in het onderzoek van Mehta et al. (2017). We zullen dus proefondervindelijk moeten ondervinden of de 2D pose estimation van VNect voldoende is, of we toch met een extra pose estimation techniek zullen moeten werken.

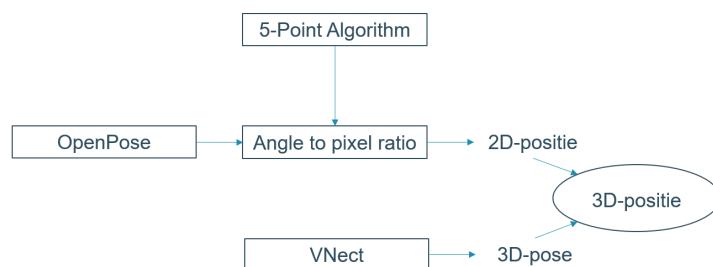
Hoofdstuk 3

Uitwerking

In dit hoofdstuk gaan we verder in op de uitwerking van de gekozen technieken. In de literatuurstudie hebben we een aantal technieken met elkaar vergeleken en besproken, die we zouden kunnen gebruiken voor de implementatie. We kozen hier telkens de techniek die ons het beste toepasbaar lijkt voor ons onderzoek. Een overzicht van de verschillende gekozen technieken is weergegeven in Figuur 3.1.

3.1 Specificaties

Voor we de implementatie van de verschillende gekozen technieken bekijken, geven we een samenvatting van de vooropgestelde specificaties van het huidige onderzoek. De eerste specificatie is het feit dat het over een leefruimte gaat. Een leefruimte is een zeer specifieke omgeving waarbij de camera's op verschillende plaatsen kunnen staan. Hierbij kan de afstand tussen de camera's variëren, afhankelijk van de plaatsen waar men een statische camera kan plaatsen. Om de positie van de persoon te bepalen moeten er zones zijn die zichtbaar zijn voor meerdere camera's.



Figuur 3.1: Overzicht te gebruiken technieken

Een tweede specificatie is dus dat onze techniek als input de beelden van meerdere camera's krijgt. Ten slotte gaan we er in dit onderzoek van uit dat er zich telkens maar één persoon in de leefruimte bevindt. We halen in de volgende secties wel aan hoe het theoretisch mogelijk gemaakt kan worden om de 3D-posities van meerdere mensen te bepalen, maar door tijdsgebrek hebben we dit echter niet kunnen implementeren.

3.2 Pose estimation

Een eerste stap in het uitwerken van de gevonden technieken is het benaderen van de 2D-pose van de persoon. In paragraaf 2.1.2.2 bespraken we reeds het OpenPose framework. We konden hieruit concluderen dat de resultaten van OpenPose nauwkeurig zijn, en dat deze techniek ook bruikbaar is indien er meerdere personen in het beeld aanwezig zijn. In paragraaf 2.3.5 bespraken we VNect, een techniek die aan de hand van slechts één camera de 3D-pose van een persoon kan bepalen. Deze techniek geeft naast deze 3D-pose ook de 2D-pose weer. In wat volgt bekijken we de implementatie van deze techniek om te controleren of de 2D-pose die door VNect wordt afgeleverd bruikbaar is voor onze toepassing. Als dit het geval is hebben we het OpenPose framework niet meer nodig om de 2D-pose te bepalen, aangezien VNect zowel de 2D- als de 3D-pose benadert.

3.2.1 VNect

Er zijn een aantal implementaties van het VNect algoritme publiek beschikbaar. Het getrainde 3D-model van de mens is echter niet publiek beschikbaar, maar wordt wel gedeeld voor onderzoeksdoeleinden. Daarom hebben we contact opgenomen met Dushyant Mehta, één van de hoofdonderzoekers van het project. Na het onderwerp van het huidige onderzoek kort geschatst te hebben, en uit te leggen waarvoor we VNect zouden willen gebruiken, kregen we toegang tot een democode en het model. De geleverde democode is geïmplementeerd in matlab en maakt gebruik van Caffe (beschreven in Jia et al. (2014)), MatCaffe (Jia et al. (2014)) en BAIR Reference Caffenet(Jia et al. (2014)). Deze laatste bevat een aantal BAIR-getrainde modellen. Bij de bijgeleverde code zijn een aantal voorbeeldafbeeldingen toegevoegd waarop de democode werd uitgevoerd. Deze voorbeeldafbeeldingen vormen een sequentie. De input met de bijhorende output is weergegeven in Figuur 3.2 en Figuur 3.3. Op Figuur 3.3 zien we dat de output van VNect bestaat uit drie delen: een 2D-pose (links), een heatmap van de verschillende gewrichten (midden) en een 3D-pose (rechts). We zien dat de 2D-pose de pose van de inputafbeelding goed benadert en dat de heatmap van de gewrichten visueel gezien klopt. De 3D-pose wordt op een andere manier weergegeven dan met een orthogonaal assenstelsel, waardoor we niet meteen visueel kunnen controleren of de 3D-pose klopt. Deze resultaten zagen er echter veelbelovend uit, waardoor we besloten om VNect uit te



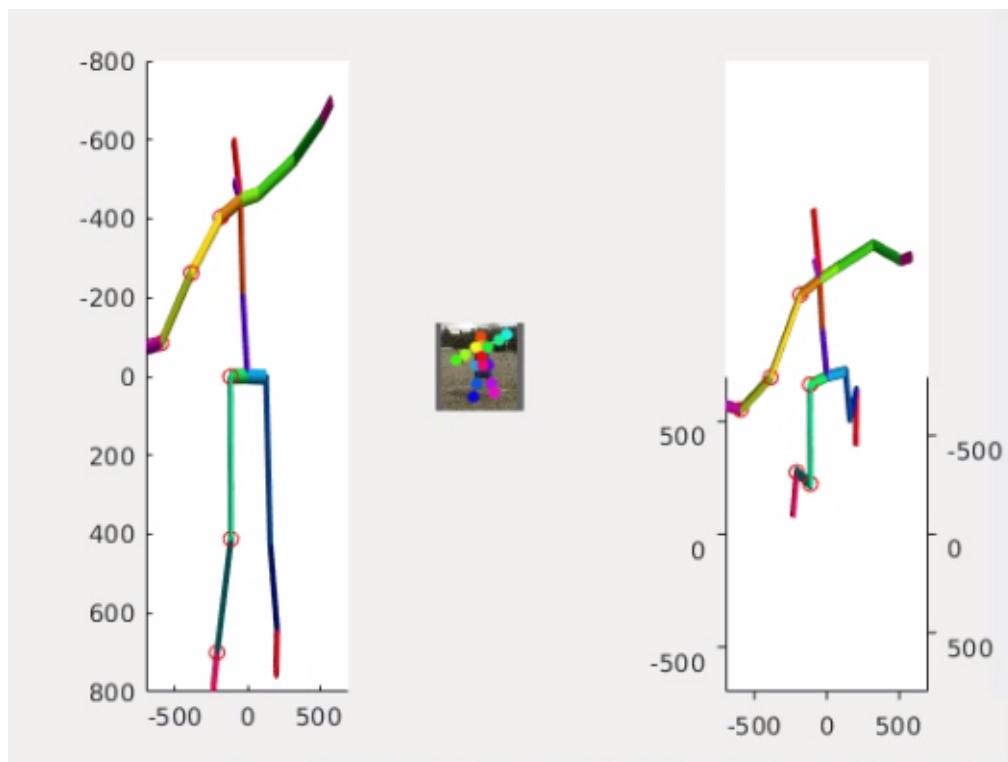
Figuur 3.2: Inputafbeeldingen van de demo van het VNect algoritme

voeren op een aantal eigen frames die gemaakt werden in een leefruimte. Op deze frames kan onder andere gecontroleerd worden of de techniek ongevoelig is voor occlusie. Op de bijgeleverde voorbeeldafbeeldingen is de persoon namelijk volledig zichtbaar.

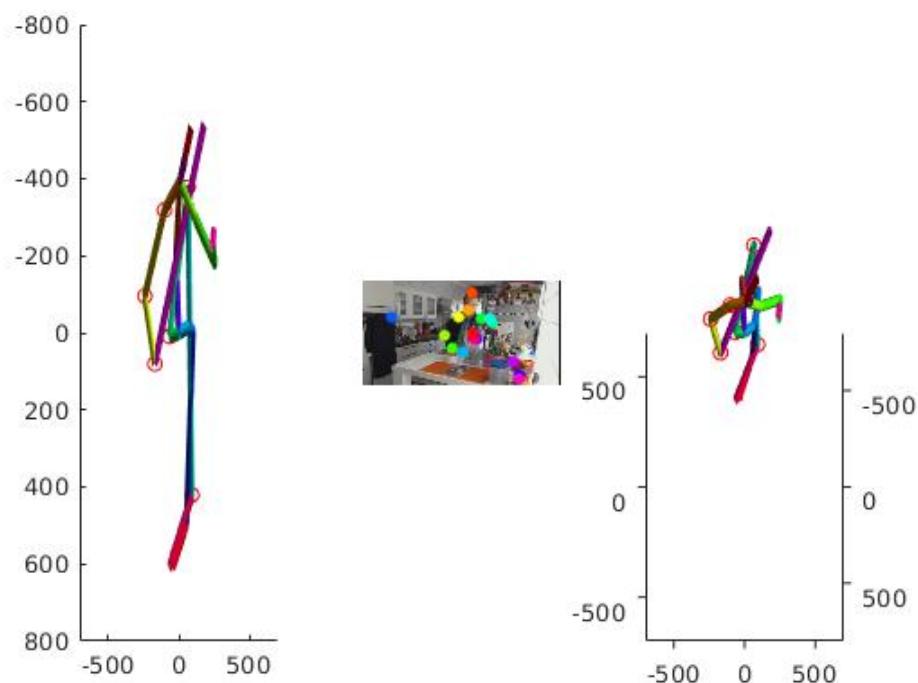
Om VNect op eigen materiaal te kunnen uittesten hebben we de democode wat aangepast. Deze werd namelijk zo geschreven dat er een reeks afbeeldingen met een bepaalde naam wordt ingelezen, waarop VNect vervolgens uitgevoerd wordt. We hebben deze democode aangepast zodat het mogelijk is om de naam van een videofile in te geven.

Het uitvoeren van de aangepaste democode van het VNect algoritme op een aantal videosequenties gaf echter niet de resultaten waarop we hadden gehoopt. Bij de bijgeleverde voorbeeldafbeeldingen konden de 2D-pose en heatmaps van de gewrichten nog visueel goedgekeurd worden, maar dat was nu niet meer het geval wanneer we de techniek uittestten op andere beelden.

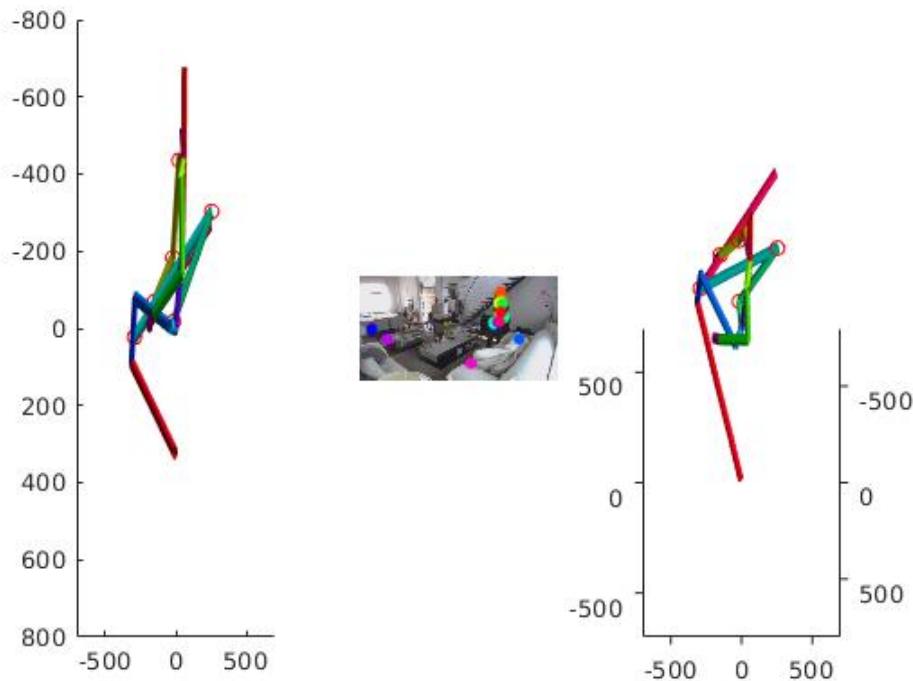
Figuur 3.4 en Figuur 3.5 geven de 2D- en 3D-pose aan voor twee voorbeeldframes uit de eigen videosequentie. We zien hierbij dat de 2D- en 3D-pose vormen aannemen die niet zijn af te leiden uit de afbeelding. Op de middelste afbeeldingen (heatmaps van de verschillende gewrichten) zien we dat het algoritme een aantal gewrichten niet juist lokaliseert en deze op verkeerde plaatsen aanduidt. Als het algoritme in een eerste fase de gewrichten niet op de juiste plaats in een afbeelding kan lokaliseren, zullen bijgevolg de 2D- en 3D-pose niet correct benaderd kunnen worden. Om de lokalisatie van de gewrichten iets beter te kunnen bestuderen geven we op afbeelding 3.6 de benadering van alle afzonderlijke gewrichten weer. Op deze afbeelding zien we dat de zichtbare lichaamsdelen zoals de nek, schouders en polsen goed gelokaliseerd worden. De gewrichten die door occlusie niet zichtbaar zijn zoals de knie, enkel en heup, worden verkeerd geschat. Het algoritme zal niet besluiten dat het gewricht onzichtbaar is, maar zal naar de plaats blijven zoeken die het best op een bepaald lichaamsdeel lijkt. Deze vaststelling verraste ons, aangezien VNect zoals beschreven in paragraaf 2.3.5 eerst de persoon detecteert aan de hand van een persoonsdetector om dan binnen de bekomen bounding box te gaan zoeken naar de verschillende gewrichten.



Figuur 3.3: Outputafbeelding van VNect met een testafbeelding als input: 2D-pose (links), heatmaps (midden) en 3D-pose(rechts)



Figuur 3.4: Ongewenste output VNect 1

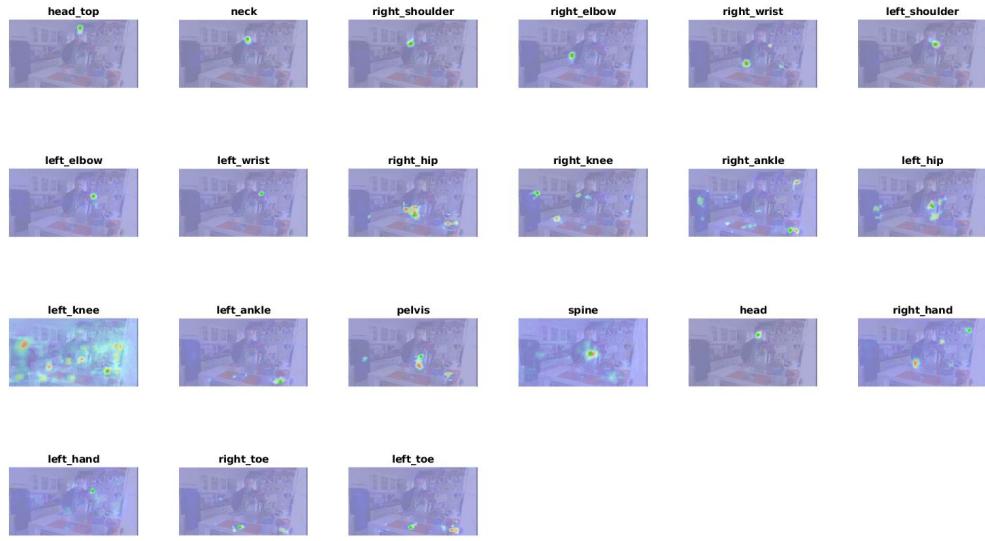


Figuur 3.5: Ongewenste output VNect 2

Naar aanleiding van dit verrassende resultaat namen we opnieuw contact op met onderzoeker Dushyant Mehta om hem in te lichten over de bekomen resultaten. Hij bevestigde dat hun algoritme niet occlusieongevoelig is, wat we reeds konden vermoeden op basis van onze resultaten. Om deze reden hebben we dan ook besloten om VNect niet te gebruiken voor dit onderzoek. Het algoritme kan in situaties zonder occlusie een nauwkeurige 3D-pose afleveren. Aangezien dit onderzoek vooral gericht is op situaties in leefruimtes is occlusie echter een veelvoorkomend fenomeen, waardoor we hier rekening mee moeten houden.

3.2.2 OpenPose

In paragraaf 3.2.1 onderzochten we of we VNect kunnen gebruiken voor de benadering van zowel de 2D-pose als de 3D-pose. Hieruit is echter gebleken dat het VNect algoritme niet kan gebruikt worden in het huidige onderzoek. Daarom onderzoeken we in dit hoofdstuk de implementatie van het OpenPose framework voor de benadering van de 2D-pose van de persoon in beeld. De verkregen keypoints kunnen we dan later gebruiken om de positie van de persoon in top-down view te benaderen.



Figuur 3.6: Heatmaps horende bij Fig 3.4

Voor ons onderzoek werd OpenPose geïnstalleerd op een Ubuntu 16.04 besturingssysteem met een GTX 1050TI grafische kaart. De vereisten voor het gebruik van OpenPose zijn onder andere de installatie van CUDA 8 en cuDNN 5.1. De NVIDIA CUDA Toolkit wordt gebruikt als ondersteuning voor applicaties die met een hoge performantie op een GPU draaien. Het bevat bibliotheken die onder andere het gebruik van deep learning en video processing toelaten. cuDNN (Chetlur et al. (2014)) is de NVIDIA CUDA Deep Neural Network library. Het is een GPU versnelde library die primitieve deep neural networks bevat. Verder maakt OpenPose ook gebruik van OpenCV en van Caffe, wat reeds besproken werd in paragraaf 3.2.1.

Door eerdere installaties beschikte het systeem reeds over CUDA 9.1 en cuDNN 7.1. Het installeren en compileren van OpenPose met deze versies van CUDA en cuDNN zorgde echter steeds voor foutmeldingen. Daarom besloten we om te werken met Docker (zoals beschreven door Bas-hari Rad et al. (2017)). Met Docker beschikken we over de mogelijkheid om een applicatie zoals OpenPose uit te voeren in een container. Deze container kunnen we dan voorzien van de bibliotheken, afhankelijkheden enzovoort die de applicatie nodig heeft, onafhankelijk van de andere installaties op het systeem. Voor ons onderzoek betekent dit dat we een docker kunnen voorzien van CUDA 8.0 en cuDNN 5.1, zonder dat hierdoor de versies op ons systeem aangetast worden. Aangezien we een applicatie willen uitvoeren die gebruikmaakt van een grafische kaart, moeten we ook NVIDIA-docker installeren. Om Docker te gebruiken moet een dockerfile gecreëerd worden, waarin de nodige commando's opgenomen worden om de specificaties te installeren in het docker

image. Het commando dat wij gebruiken om de docker op te starten is:

```
nvidia-docker run -rm -it -e DISPLAY=$DISPLAY -v home/jeroen/Documents/Masterproef/openpose_output/camera1:/usr/local/openpose/output -v /tmp/.X11-unix:/tmp/.X11-unix openpose /bin/bash
```

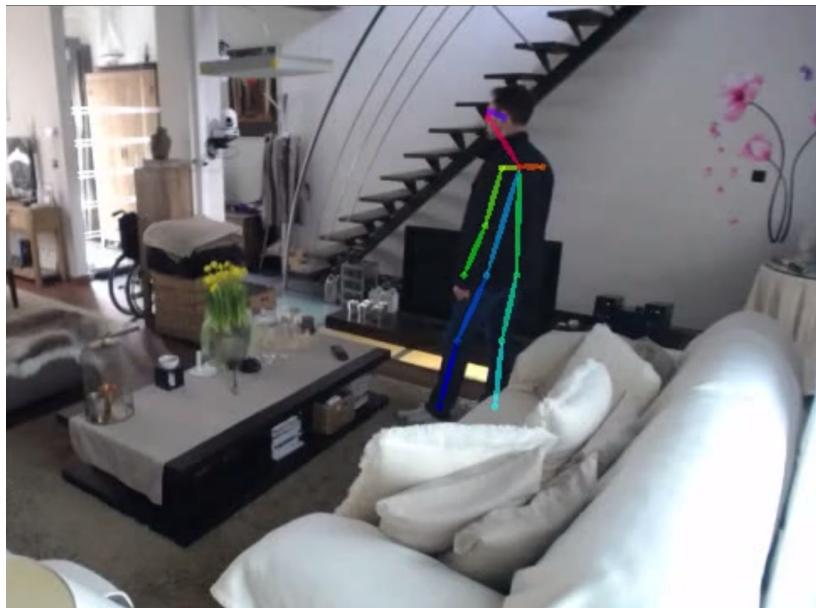
Dit commando moet worden uitgevoerd in de directory waarin de dockerfile staat met de informatie over de vereiste specificaties voor het opstarten van de docker. De dockerfile plaatsen we in de directory van OpenPose. Op deze manier wordt er een tijdelijke docker image aangemaakt waarin een docker container wordt gemaakt. Bij het sluiten van de image wordt de bijhorende container ook verwijderd. Daarom mounten we een aantal directories door middel van het -v argument. De eerstgenoemde directory bestaat op het werkelijke toestel. Deze wordt gevolgd door de directory binnen de image docker. Op die manier kunnen we output gegenereerd door OpenPose en opgeslagen in de docker: '/usr/local/openpose/output' als het ware kopiëren naar een map op het werkelijke toestel: home/jeroen/Documents/Masterproef/openpose_output/camera1, zodat na het sluiten van de docker image de output van OpenPose toch bewaard blijft. Eenmaal de docker image opgestart is, kunnen we OpenPose opstarten zoals we het zonder een docker zouden doen. OpenPose heeft heel wat verschillende uitvoeringsmodi die geactiveerd kunnen worden door het activeren van een aantal vlaggen. De standaarduitvoering van OpenPose gebeurt door middel van het commando:

```
./build/examples/openpose/openpose.bin
```

Op deze manier wordt OpenPose gestart met als input de gegevens verkregen via webcam. Er is echter ook de mogelijkheid om afbeeldingen of video's als input te gebruiken. Dit kan aangegeven worden door middel van -video of -image_dir, telkens gevolgd door de directory met de video of afbeelding waarop men het OpenPose framework wil uitvoeren. Verder hebben we nog de keuze om enkel de 2D-pose van de persoon te benaderen, of ook het gezicht en de handen. Dit kunnen we aangeven door het stellen van een aantal vlaggen door -face of -hand aan het bovenstaande commando toe te voegen. Op dezelfde manier zijn er ook verschillende vlaggen die gebruikt kunnen worden voor het opslaan van het resultaat van OpenPose. Een aantal voorbeelden zijn: write_images, write_video en write_json, write_keypoint_json. Om het resultaat visueel te kunnen controleren kozen we voor de vlag -write_video. Om ook de coördinaten van de verschillende keypoints als resultaat te krijgen, maakten we gebruik van de vlag -write_keypoint_json.

```
./build/examples/openpose/openpose.bin -video output/1.mp4 -write_video output/result.avi  
-write_keypoint_json output/
```

Figuur 3.7 toont de benadering van de 2D-pose van de persoon op hetzelfde frame als Figuur 3.5. Op basis van een visuele controle kunnen we besluiten dat dit inderdaad een goede benadering is van de 2D-pose van de persoon, hoewel een deel van het been geoccludeerd is. In eerste instantie

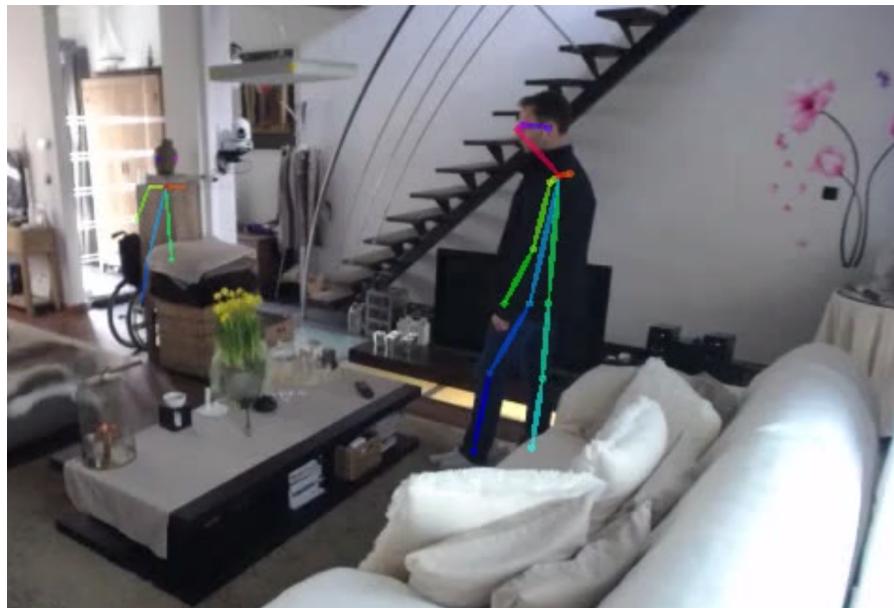


Figuur 3.7: OpenPose output

waren er echter ook een aantal sequenties waarin het OpenPose framework op enkele plaatsen de pose van een persoon benaderde waar er geen persoon aanwezig was. Een voorbeeld van zo een afbeelding is Figuur 3.8.

We kunnen dit probleem echter verhelpen door een aantal parameters in het bestand 'openpose/examples/openpose/openpose.cpp' aan te passen. Zo hebben we de double 'render_threshold'. Dit is een thresholdwaarde die ingesteld kan worden voor de poses, de handen en de gezichten van de personen in beeld waarbij enkel de keypoints die een hogere score hebben dan de ingestelde threshold zullen gevisualiseerd worden. Standaard staat deze waarde ingesteld op 0,05%, wat een zeer lage threshold is. Afhankelijk van het doel waarvoor het OpenPose framework gebruikt wordt, zal dit op een andere waarde ingesteld worden. Als we de pose willen benaderen van de lichaamsdelen die duidelijk zichtbaar zijn, dan moeten we een hogere threshold instellen dan 0,05%. Dit betekent dat OpenPose niet zelf een benadering zal geven van de pose van geoccludeerde lichaamsdelen. Indien men dit wel wil, kan men de threshold eerder een lagere waarde geven zoals 0,1%. Aangezien er dan meer moet geraden worden naar de positie van de verschillende lichaamsdelen zal dit uiteraard ook resulteren in meer false positives zoals weergegeven in Figuur 3.8.

Aangezien het voor onze applicatie niet nodig is om de volledige pose te kennen, maar slechts een aantal punten volstaan, is het beter om de threshold te verhogen. Op die manier weten we dat de verkregen positie van de persoon de effectieve positie is, en dat er niet een aantal geocludeerde punten geschat zijn.



Figuur 3.8: OpenPose fout

Zoals reeds vermeld kan het resultaat opgeslagen worden onder de vorm van Json files. Als de write.json vlag gebruikt wordt, wordt de pose-data van de verschillende personen opgeslagen aan de hand van een JSON writer. Indien de vlaggen voor het benaderen van het gezicht en de handen gebruikt worden, worden ook deze poses weergegeven in de JSON files. Elke JSON file bevat een array van personen waarbij elk persoonobject is onderverdeeld in verschillende deelobjecten overeenkomend met de poses van de gevraagde lichaamsdelen: pose_keypoints_2d, face_keypoints_2d, enzovoort. Elk deelobject bestaat dan uit gegevens van de verschillende keypoints behorende tot dat object. Zo wordt er van elk keypoint een x- en y-coördinaat gegeven, in combinatie met een betrouwbaarheidsscore. Deze score kan dan later gebruikt worden om later enkel de keypoints in te lezen die boven een bepaalde thresholdwaarde liggen. Een voorbeeld van een Json-file wordt weergegeven in Figuur 3.9.

3.3 5 Point Algorithm

In 3.2 bekeken we de implementatie van het VNect algoritme en concludeerden we dat we deze techniek niet kunnen gebruiken voor onze toepassing. Vervolgens hebben we de implementatie van OpenPose bekeken. In de literatuurstudie besloten we dat we de punten van een 2D-pose techniek willen gebruiken voor het plaatsen van de persoon in top-down view. Indien we weten hoe elke persoon zich ten opzichte van een bepaalde camera positioneert in een vlak evenwijdig aan de camera (zonder diepte-informatie), moeten we echter nog weten hoe de camera's ten opzichte van elkaar gepositioneerd zijn om de werkelijke positie van de persoon in de ruimte te kunnen bepalen.

```
{
  "version":1.2,"people":[
    {"pose_keypoints_2d": [
      0,0,0,
      435.832,49.4192,0.867684,
      448.513,50.3738,0.855272,
      449.498,67.0481,0.845401,
      449.49,82.6957,0.681943,
      426.047,49.374,0.906102,
      419.195,66.0183,0.746369,
      423.083,81.6844,0.453648,
      441.636,86.5691,0.768263,
      442.63,109.087,0.778727,
      439.721,135.468,0.676801,
      426.05,84.627,0.842534,
      426.051,112.024,0.852858,
      425.045,137.474,0.796706,
      0,0,0,
      0,0,0,
      445.588,35.7132,0.758453,
      434.848,34.7393,0.736995],
      "face_keypoints_2d": [],
      "hand_left_keypoints_2d": [],
      "hand_right_keypoints_2d": [],
      "pose_keypoints_3d": [],
      "face_keypoints_3d": [],
      "hand_left_keypoints_3d": [],
      "hand_right_keypoints_3d": []
    ]
  ]}
]
```

Figuur 3.9: Output OpenPose JSON

In paragraaf 2.2.1 hebben we verschillende kalibratietechnieken met elkaar vergeleken en kwamen we tot de conclusie dat kalibratie aan de hand van een dambordpatroon voor het huidige onderzoek minder geschikt is. De camera's in een leefruimte kunnen op een grote afstand van elkaar staan waardoor de overlap tussen beide camera's kleiner wordt. We vonden echter een techniek die de externe kalibratieparameters bepaalt aan de hand van vijf zelf gekozen punten die in beide cameraviews aangeduid worden: het 5-point algorithm (zie paragraaf 2.2.1.3). In tegenstelling tot VNect is er echter geen officiële implementatie beschikbaar van deze techniek, maar wel een aantal publieke implementaties. Daarom kozen we voor een publieke implementatie. Deze implementatie vertrekt van een gegeven translatie en rotatie van een view ten opzichte van een tweede view. Vervolgens worden er een aantal punten gegenereerd die gebruikt worden om het algoritme te testen.

Na uitvoering van het algoritme krijgen we inderdaad de gegeven translatie en rotatie. De democode lijkt dus te werken. Uiteraard kunnen we voor onze toepassing niet vertrekken van een gegeven translatie en rotatie, aangezien we net naar deze gegevens op zoek zijn. Daarom hebben we de democode aangepast zodat de gebruiker zelf een aantal punten kan ingeven. Indien we zelf een set van vijf punten ingeven krijgen we echter geen output. Indien we er een extra punt aan toevoegen, krijgen we wel een output. De oorzaak hiervan is waarschijnlijk dat RANSAC pas gebruikt wordt vanaf zes punten, en niet vanaf vijf. Om een eerste test uit te voeren of het algoritme effectief werkt hebben we hard-coded twee sets van exact dezelfde punten ingegeven. In theorie zou dit overeen moeten komen met twee camera's die op exact dezelfde plaats staan. In werkelijkheid krijgen we echter toch een translatie en rotatie als resultaat zoals weergegeven op

```
RANSAC
[-6.737039866254233e-17, -0.07066138042785018, 0.3867250583298972;
 0.07066138042785011, 9.74334773964425e-17, -0.587750541105705;
 -0.3867250583298973, 0.587750541105702, 2.59635506749254e-17]=====
5-pt-nister rvec:
[0;
 0;
 0]
[2.6113068509935;
 1.718174163412199;
 0.313940241489612]
5-pt-nister tvec:
[-0.831204786530694;
 -0.5469118223996668;
 -0.09993028253707038]
[0.831204786530694;
 0.5469118223996668;
 0.09993028253707038]
```

Figuur 3.10: Output 5 Point Algorithm

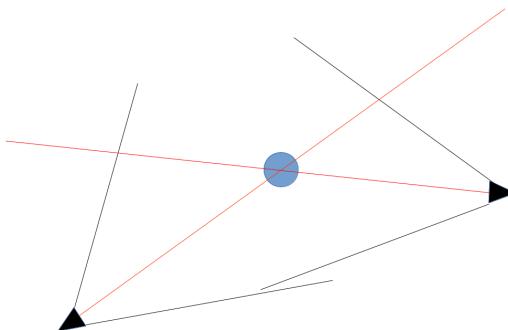
Figuur 3.10 Aangezien de oorspronkelijke democode wel leek te werken is het mogelijk dat we met een aantal extra stappen wel het beoogde resultaat zouden kunnen bekomen. Door de beperkte tijd van dit onderzoek hebben we echter besloten om dit algoritme niet te debuggen maar dit op te lossen door middel van een eigen kalibratiemethode, beschreven in paragraaf 3.4.3.1.

3.4 Top-down positie

In dit hoofdstuk bespreken we hoe we de top-down positie van een persoon in een leefruimte zullen bepalen. Eerst bespreken we verschillende mogelijke situaties waarin de persoon in beeld zich kan bevinden. Vervolgens bespreken we een eigen implementatie om de positie van de persoon te benaderen in deze verschillende situaties.

3.4.1 Theoretische uitwerking

In sectie 2.2.2 hebben we reeds gezien dat de positie van een persoon in beeld benaderd kan worden door een lijn. In dit hoofdstuk werken we dit verband verder uit voor de verschillende situaties waarin de persoon zich zou kunnen bevinden. De angle-to-pixel methode vertrekt van twee afbeeldingen die op hetzelfde moment door twee verschillende camera's genomen zijn, en waar de persoon op te zien is. Indien we de persoon op één camera in 2D kunnen lokaliseren, kunnen we (in top-down view) een rechte trekken van de camera naar de persoon. De persoon bevindt zich dan bij benadering op deze rechte. Een nauwkeurigere benadering kan gevonden worden door de persoon op een tweede beeld te lokaliseren en ook een rechte te trekken van de tweede camera naar de persoon. Deze twee lijnen zullen elkaar snijden en geven zo bij benadering



Figuur 3.11: Positioneren in top-down

de positie van de gedetecteerde persoon weer. Deze methode wordt gevisualiseerd in Figuur 3.11. Hierbij worden aanvankelijk twee beperkingen opgelegd.

1. In onze beelden is er telkens maar één persoon te zien. Indien er twee personen in de ruimte aanwezig zijn kunnen we niet weten waar deze personen zich bevinden. Dan zijn er in een oneindige ruimte namelijk vier snijpunten, waarbij het niet mogelijk is te definiëren welke twee van de vier punten nu de personen voorstellen. Dit probleem stelt zich ook indien er meer dan twee personen aanwezig zijn. Wanneer er slechts twee personen in beeld zijn kan dit probleem wel opgelost worden door een zone te bepalen waarin beide personen op hetzelfde moment op drie camera's zichtbaar zijn. Hiertoe zijn we in de beperkte tijdspanne van dit onderzoek echter niet geraakt.
2. De camera's kijken niet neer op de persoon. Dit wil zeggen dat ze niet zoals bewakings-camera's aan het plafond hangen en vanuit de hoogte op de mens neerkijken. Indien dit wel het geval zou zijn, zou een lijn tussen de lens van de camera en de persoon door de grond getrokken worden. Hierdoor verliezen we een deel van de informatie, zoals verder in dit hoofdstuk duidelijk wordt.

Het resultaat van OpenPose is een set van 15 of 18 punten (afhankelijk van het gekozen output type: MPI body parts of COCO body parts) die de pose van de persoon beschrijven (Zie Figuur 2.5).

Het resultaat van OpenPose is een set van x- en y-coördinaten van de punten, samen met een score die de kans weergeeft dat een punt tot de persoon behoort. Voor het schatten van de positie van de persoon hebben we echter niet al deze punten nodig. Voor de implementatie van de angle-to-pixel methode gebruiken we de coördinaten van de nek. Dit is een lichaamsdeel met als voordeel dat het in vele toestanden zichtbaar is. Indien de persoon aan tafel zit of achter een meubel staat, zijn lichaamsdelen zoals de benen en heupen namelijk niet zichtbaar. De nek wel. Een ander

voordeel is de nauwkeurigheid. In tabel 2.1 zien we dat het hoofd een nauwkeurigheid heeft van 91.2%, wat hoger ligt dan andere lichaamsdelen.

De beelden die we gebruiken om onze methode te testen, werden gefilmd door camera's met een view van 78,6 graden. De beelden zelf zijn gemaakt met een bepaalde resolutie, in ons geval 1920x1080px. Dit betekent dat de breedte gelijk is aan 1920 pixels en de hoogte aan 1080 pixels. Indien we de verhouding nemen tussen de view van de camera en de breedte van het beeld uitgedrukt in het aantal pixels (zie Formule 3.1), hebben we een getal dat het aantal graden per pixel voorstelt.

$$\frac{\Delta\theta}{\Delta px} = \frac{angleofview(^{\circ})}{width(px)} \quad (3.1)$$

Dit zou in ons geval uitkomen op:

$$\frac{78.6^{\circ}}{1080px} = 0.0728 \frac{^{\circ}}{px}$$

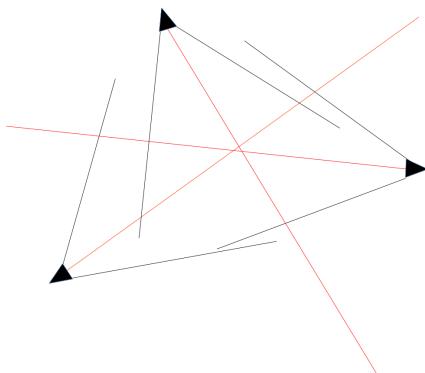
Aangezien OpenPose voor elk van de punten in de set van de pose de x-en y-waarde geeft, is een simpele vermenigvuldiging van het resultaat van deze verhouding met de x-waarde van het gekozen punt voldoende om de hoek te weten tussen de linker zijde van het field of view en het gekozen punt. Om de angle-to-pixel methode uit te voeren moeten we natuurlijk de onderlinge posities van de camera's kennen.

3.4.2 Verschillende gevallen hoek-per-pixelmethode

Natuurlijk zullen we ons niet altijd in een situatie bevinden waarbij een persoon zichtbaar is voor meerdere camera's. Afhankelijk van de positie waarin de persoon zich bevindt zal er een andere methode gebruikt moeten worden. Zo onderscheiden we hieronder achtereenvolgens de situaties waarin de persoon zichtbaar is voor geen, één en twee of meer camera('s).

3.4.2.1 Twee of meer camera's

Zoals reeds besproken werd in het vorige hoofdstuk kunnen we al een schatting maken van de positie van de persoon in top-down view met behulp van twee camera's. Indien er meer dan twee camera's aanwezig zijn, kunnen deze dienen ter controle. Voor elke camera zal er namelijk een projectielijn komen tussen de camera en de persoon. Bij de aanwezigheid van een derde camera gebruiken we dezelfde werkwijze als wanneer er twee camera's gebruikt worden. In theorie zal de lijn tussen de persoon en de cameralens dan door het snijpunt van de andere twee lijnen lopen. Door de aanwezigheid van fouten is het echter ook mogelijk dat de derde lijn de andere twee niet zal snijden in het snijpunt (Figuur 3.12). Door de snijpunten van deze drie projectielijnen wordt er een driehoek gevormd waarbinnen de persoon zich zou moeten bevinden. We kunnen het centrum



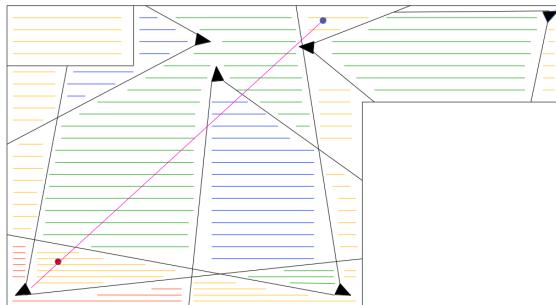
Figuur 3.12: Drie camera's: niet hetzelfde snijpunt

van deze driehoek berekenen om met een grotere nauwkeurigheid de positie van de persoon te bepalen.

3.4.2.2 Één camera

In het geval dat de persoon maar in beeld is voor één camera, moeten we onze methode aanpassen. We kunnen dan maar één lijn trekken tussen de camera en de persoon in de scène. Het enige wat hierbij geweten is, is dat de persoon zich bij benadering op deze lijn bevindt. Waar precies op deze lijn, kan men op het eerste zicht niet weten. We kunnen echter wel plaatsen beginnen uitsluiten, waar de persoon zich niet kan bevinden. Figuur 3.13 toont een voorbeeld van een setup van zes camera's in een leefruimte. De views van de verschillende camera's worden door middel van lijnen weergegeven. De plattegrond is onderverdeeld in verschillende zones waarvan de grenzen gevormd worden door de views van de camera's. Elke zone heeft hierbij een kleur gekregen, afhankelijk van het aantal camera's waarvoor zone zichtbaar is. De kleuren rood, geel, groen en blauw duiden achtereenvolgens zones aan die zichtbaar zijn voor geen, één, twee en drie camera's. Indien een persoon zich in een gele zone bevindt is hij dus slechts zichtbaar op één camera en kunnen we maar één projectielijn tekenen waarop de persoon zich moet bevinden. Om meer informatie over zijn positie te weten kunnen we zoals eerder gezegd werken met uitsluiting of eliminatie van zones. Als we terugkeren naar Figuur 3.13 zien we een persoon (rood) in een zone die zichtbaar is voor één camera. Volgens onze techniek moet de persoon zich bevinden op de roze lijn. We kunnen alle zones op deze lijn uitsluiten die zichtbaar zijn voor meerdere camera's. Moest de persoon zich namelijk in zo een zone bevinden, zou hij op meerdere camera's zichtbaar moeten zijn. Hierdoor kunnen we het aantal zones waarin de persoon zich zou kunnen bevinden al verkleinen.

Er zijn echter situaties denkbaar waarin de persoon zich dan nog in verschillende zones zou kunnen



Figuur 3.13: Voorbeeld van een plattegrond. Rood, Geel, Groen, Blauw betekenen respectievelijk: Persoon zichtbaar op geen, één, twee of drie camera's

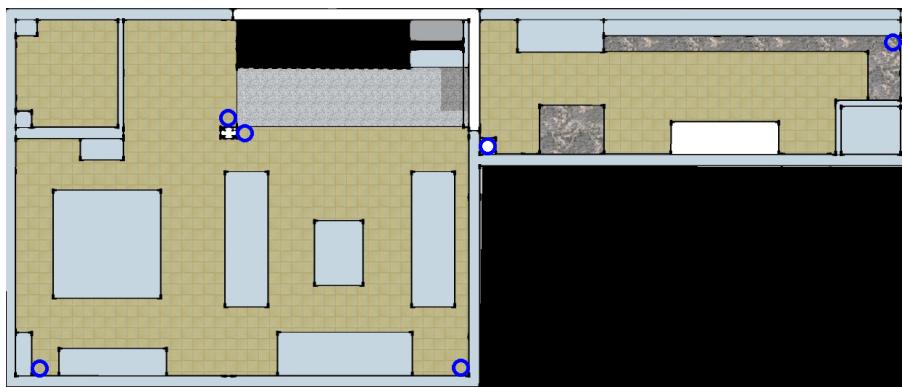
bevinden. Zo weten we met behulp van onze techniek niet of de persoon zich op de blauwe of rode stip bevindt. Het creëren van een geschiedenis van de posities kan hier een oplossing voor zijn. Een tweede oplossing voor deze situatie zou kunnen zijn om de grootte van de persoon in rekening te brengen. We kunnen dit op voorhand ingeven of een gemiddelde grootte van personen gebruiken. Indien we kalibratie kunnen toepassen, kunnen we de grootte ook bepalen aan de hand van triangulatie in een zone met twee of meer camera's. Als we de grootte van de persoon kennen, kunnen we een schatting maken van de diepte van deze persoon ten opzichte van de camera.

3.4.2.3 Geen camera's

De situatie waarin een persoon voor geen enkele camera zichtbaar is, is uiteraard de minst gunstige. In dat geval vermindert de nauwkeurigheid nog meer. Afhankelijk van het aantal 'onzichtbare' zones in de leefruimte kunnen we echter wel nog een schatting maken van zijn of haar positie. Indien de persoon op geen enkel beeld zichtbaar is, kunnen we alle zones behalve de rode zones uitsluiten. Hierdoor blijft er maar een zeer beperkte zone over waarin de persoon zich kan bevinden. Ook hier kan een geschiedenis van de posities een oplossing bieden om een nauwkeurigere schatting te maken van de positie. Uiteraard moeten er voldoende camera's zijn om de positie van de persoon ook te kunnen schatten in een zone die slechts door één camera of zelfs helemaal niet zichtbaar is. Er moeten dus voldoende zones zijn die voor twee of meer camera's zichtbaar zijn, om de nauwkeurigheid van het systeem te waarborgen.

3.4.3 Implementatie

Op basis van de boven genoemde theorie werkten we zelf een implementatie uit. Deze werd geschreven in C++ en maakt gebruik van OpenCV3. Bij deze implementatie is het de bedoeling dat de gebruiker zelf de positie en rotatie van de verschillende camera's zal bepalen, om op deze ma-



Figuur 3.14: Posities van de verschillende camera's

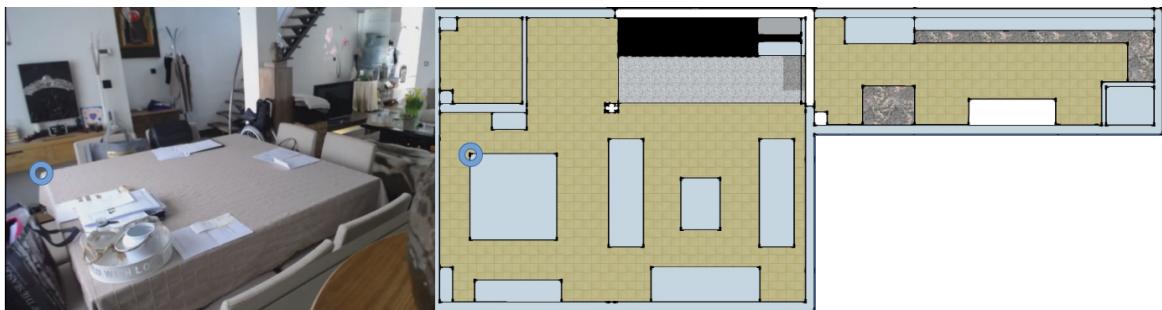
nier het gebrek aan een kalibratietechniek op te vangen. Vervolgens zal de top-down positie van de persoon in de leefruimte benaderd worden door gebruik te maken van de gekende posities van de verschillende camera's.

3.4.3.1 Kalibratie

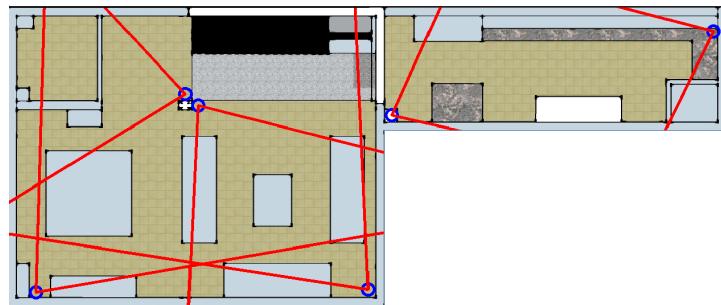
In eerste instantie wordt er aan de gebruiker gevraagd om de randpunten van de leefruimte aan te klikken. Deze punten worden gebruikt om een masker te maken, dat in de volgende stappen over de afbeelding geplaatst wordt indien we punten of projectielijnen tekenen op de afbeelding. Zo wordt de kaart van de leefruimte overzichtelijk weergegeven. Vervolgens kan de gebruiker ingegeven hoeveel camera's hij wil positioneren. Nadat de gebruiker dit aantal heeft ingegeven kan hij zelf op het grondplan van de ruimte de camera's plaatsen. De translatie van de verschillende camera's ten opzichte van elkaar wordt dus reeds benaderd. Een voorbeeld van een leefruimte waarin de posities van zes camera's aangeduid zijn, wordt weergegeven in Figuur 3.14.

Indien we de positie van de persoon in de leefruimte willen bepalen, moeten we echter ook weten hoe de camera's ten opzichte van elkaar geroteerd staan. Dat betekent dat we de linker- en rechterzijde van de angle of view te weten moeten komen. Hiervoor maken we gebruik van de wetenschap dat de angle of view van alle camera's een constante is. We kennen de rotatie van een camera indien we weten welk punt op een beeld van de camera overeenkomt met een punt op de plattegrond van de leefruimte. Er bestaat namelijk maar één rotatie die aan deze voorwaarde voldoet. We hebben dit geïmplementeerd door gebruik te maken van de hoek-per-pixel methode. Voor het positioneren van de camera hebben we telkens twee afbeeldingen nodig: het grondplan en het beeld van de camera. Per camera moeten we telkens een punt aanduiden op het camerabeeld samen met het overeenkomstige punt op het grondplan.

In Figuur 3.15 is een voorbeeld weergegeven waarin een camera gekalibreerd wordt door het aanduiden van dezelfde hoek van de tafel op het camerabeeld als op het grondplan. Vervolgens wordt



Figuur 3.15: Kalibreren van camera's door bepalen van puntenkoppels



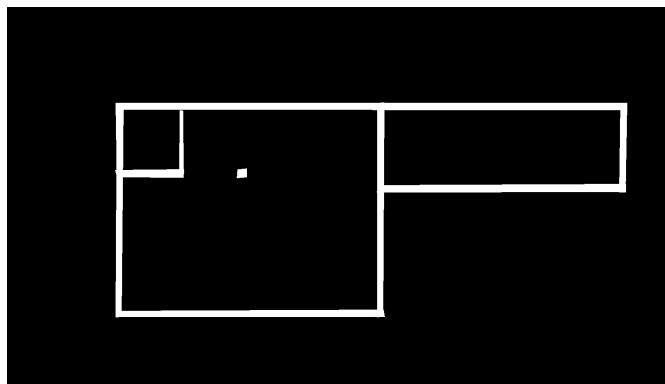
Figuur 3.16: Positie + rotatie van de verschillende camera's

er een rechte berekend tussen de camera (die de gebruiker reeds in een eerdere fase geïnformeerd had) en het aangeklikte punt op het grondplan.

Van deze rechte op het grondplan wordt de hoek berekend met de positieve horizontale (α). Vervolgens gebruiken we de x-coördinaat van het punt dat op het camerabeeld aangeduid is. We weten welke hoek dit punt maakt met de linkerzijde van de angle of view. Deze hoek kunnen we namelijk berekenen door de hoek-per-pixel verhouding te vermenigvuldigen met de verkregen x-positie van het aangeklikte punt (β).

De linkerzijde van de angle of view kunnen we ten slotte berekenen door β af te trekken van α . De rechterzijde wordt eenvoudig berekend door de constante van de angle of view te verminderen met β en dit resultaat op te tellen bij α . Deze berekening wordt uitgevoerd voor alle aanwezige camera's waardoor we een handmatige kalibratie uitgevoerd hebben. Op deze manier weten we hoe de verschillende camera's getransleerd en geroteerd zijn ten opzichte van elkaar. Het resultaat van het bepalen van de verschillende puntenkoppels is weergegeven in Figuur 3.16.

Om de nauwkeurigheid van de handmatige kalibratie te vergroten, hebben we besloten om de gebruiker niet slechts één koppel van punten per camera te laten aanduiden, maar drie. Vervolgens berekenen we het gemiddelde van deze drie rotaties, wat het resultaat nauwkeuriger maakt.



Figuur 3.17: Voorbeeld van een masker dat de muren aanduidt

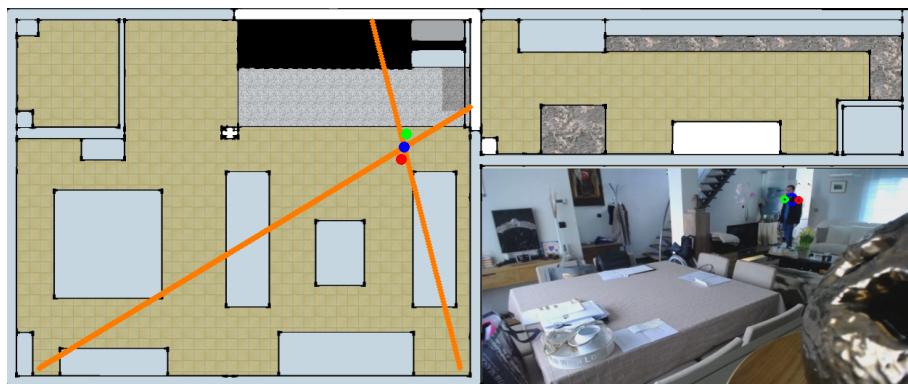
3.4.3.2 Hoek-per-pixel methode

Eens de kalibratie van de verschillende camera's uitgevoerd is, kan de positie van de persoon in top-down view benaderd worden. Om deze positie te bepalen hebben we eerst een masker gemaakt dat de muren en hoge voorwerpen van de leefruimte bevat. Dit masker wordt vervolgens op de kaart van de leefruimte geplaatst om aan te duiden dat de camera's niet door deze muren en hoge voorwerpen heen kunnen kijken. Het masker voor onze leefruimte is weergegeven in Figuur 3.17.

In paragraaf 3.2.2 hebben we reeds vermeld dat we OpenPose gebruiken om de 2D-pose van de persoon in beeld te bepalen. We hebben er ook voor gekozen om het resultaat op te slaan onder de vorm van JSON-files. Vervolgens maakten we een functie die voor alle camera's de JSON-files inleest die horen bij een bepaald framenummer. De ingelezen gegevens worden daarna opgeslagen in een vector. Vervolgens wordt per camera ook de hoek bijgehouden die het keypoint van de nek maakt met de linkerzijde van het cameraview. Deze hoek bekomen we door de x-coördinaat van dit keypoint te vermenigvuldigen met de angle-to-pixel ratio.

In eerste instantie gebruikten we enkel het keypoint van de nek om de persoon in top-down view te lokaliseren. Om extra informatie over de kijkrichting van de persoon te bekomen, hebben we dit echter uitgebreid met de keypoints van de linker- en rechterschouder. Op die manier weten we hoe de persoon gedraaid is, telkens als de persoon zichtbaar is voor twee of meer camera's.

De absolute hoek van de linkerzijde van de angle of view hebben we reeds berekend tijdens de kalibratie. In paragraaf 3.4.1 verklaren we de theoretische uitwerking van deze techniek verklaard. Daaruit kunnen we afleiden dat de volgende stap het vormen van een projectielijn tussen de camera en de persoon is. Hiervoor maken we gebruik van de vector die voor elke camera de hoeken (a) bevat die de x-coördinaat van de gekozen punten (nek, linker- en rechterschouder) maakt met de linkerzijde van de angle of view. De absolute hoek van de linkerzijde van de angle of view (b) hebben we reeds berekend tijdens de kalibratie. De absolute hoek van de projectielijn kunnen we



Figuur 3.18: Situatie waarin de persoon zichtbaar is voor twee camera's. Oranje lijnen zijn projectielijnen. De persoon wordt weergegeven door drie cirkels: groen (rechterschouder), blauw (nek) en rood (linkerschouder)

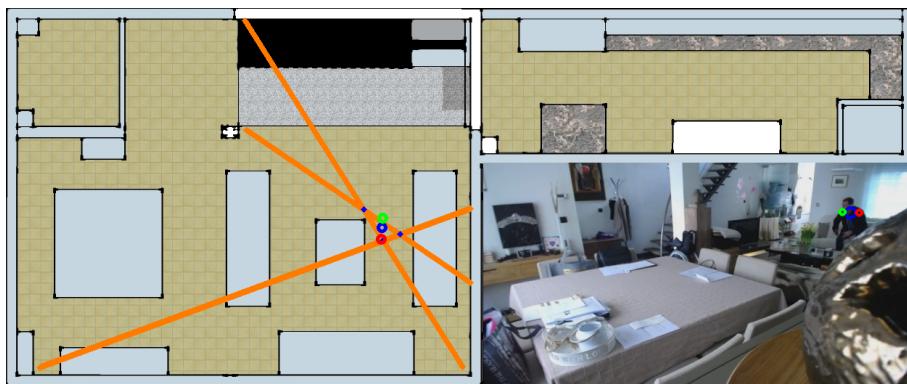
dus berekenen door het sommeren van deze twee hoeken ($a + b$). Er wordt ook bijgehouden voor hoeveel camera's de persoon juist zichtbaar is. Voor elk van deze camera's berekenen we dan de vergelijking van de projectielijn. Vervolgens bekijken we voor elk punt op deze lijn, vertrekende van de camera, of het punt in een muur gelegen is. Dit weten we door gebruik te maken van het masker (Figuur 3.17). Wanneer we een punt op de lijn tegenkomen dat in een muur gelegen is, hoeven we niet verder te kijken en weten we tot waar de camera kan kijken. Het limiteren van deze lijnen is belangrijk. Indien we dit niet zouden doen, zou de persoon zich volgens het algoritme in een kamer kunnen bevinden die in de praktijk niet zichtbaar is voor de camera.

De volgende stappen die het algoritme zet zijn afhankelijk van het aantal camera's waarvoor de persoon in de leefruimte zichtbaar is. In wat volgt bespreken we de verschillende mogelijkheden.

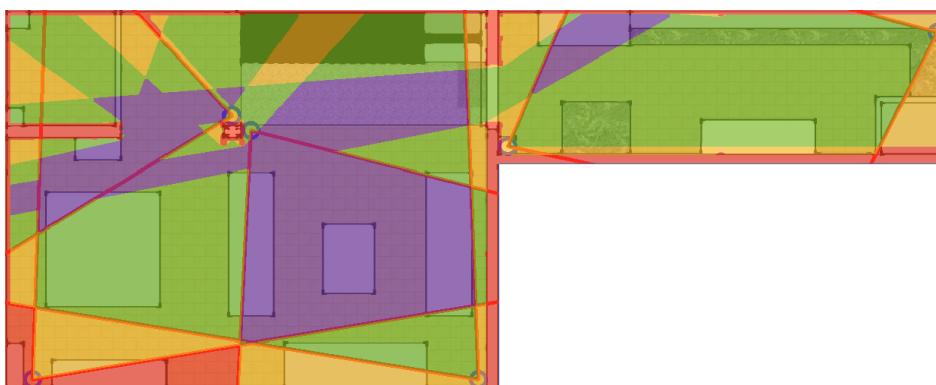
Twee camera's De eenvoudigste situatie om uit te werken is de situatie waarin de persoon maar zichtbaar is voor twee camera's. We kunnen de vergelijking van een rechte namelijk opstellen aan de hand van twee punten: de positie van de camera en het punt tot waar de camera kan kijken. Vervolgens wordt het snijpunt van deze twee rechten berekend, zoals weergegeven in Figuur 3.18.

Drie camera's Indien de persoon zichtbaar is voor drie camera's, vormen de snijpunten van de drie projectielijnen een driehoek. We hebben besloten dat we de positie van de persoon in dat geval zullen benaderen door het centrum van deze driehoek te berekenen, wat berekend kan worden door het gemiddelde van de drie punten te nemen. Een voorbeeld is weergegeven in Figuur 3.19.

Eén camera Indien de persoon slechts zichtbaar is voor één camera, weten we enkel dat de persoon zich op de projectielijn bevindt. We kennen zijn of haar exacte positie in dat geval niet, aangezien we geen snijpunt kunnen berekenen met een andere projectielijn. Het is echter wel mo-



Figuur 3.19: Situatie waarin de persoon zichtbaar is voor drie camera's



Figuur 3.20: Matrix die de verschillende zones aanduidt. Rood: zichtbaar voor geen enkele camera, geel: zichtbaar voor één camera, groen: zichtbaar voor twee camera's en paars: zichtbaar voor drie of meer camera's

gelijk om zones te elimineren waarin de persoon zich al zeker niet kan bevinden. Op de projectielijn zullen er namelijk punten zijn die ook zichtbaar zijn voor andere camera's. Aangezien de persoon niet zichtbaar is voor andere camera's, kan hij zich dus niet op deze punten bevinden.

Deze situatie hebben we aangepakt door een matrix op te stellen die voor elke pixel aangeeft voor hoeveel verschillende camera's deze pixel zichtbaar is. Een voorbeeld van zo een matrix is weergegeven in Figuur 3.20. Indien de persoon dus maar zichtbaar is voor één camera, kan hij zich niet in een zone bevinden die zichtbaar is voor meerdere camera's. We kunnen deze zones uitsluiten door op de matrix te kijken welke pixels tot welke zone behoren.

Hoofdstuk 4

Evaluatie

Om te kunnen aantonen dat onze implementatie werkt, moeten we hem eerst valideren. Dit doen we op twee manieren, namelijk op basis van een aantal top-down posities die we zelf bepaald hebben op eigen beelden en op basis van de CMU-dataset (vermeld in paragraaf 2.2.5)

4.1 Eigen data

Voor een aantal frames bepaalden we zelf de top-down positie van de persoon in de leefruimte. Dit hebben we zowel gedaan voor de situatie waarin de persoon in beeld is voor twee camera's als voor drie camera's, en dit telkens voor een 15-tal frames. Deze methode is geen nauwkeurige manier om onze techniek te evalueren aangezien we zelf de positie benaderen en niet nauwkeurig kunnen bepalen, maar het geeft ons wel een eerste idee van de ordegrootte van de fout. Over alle geëvalueerde frames heen hebben we het grootste, het kleinste en het gemiddelde verschil berekend tussen de door ons aangeduid positie en de positie zoals benaderd door onze implementatie. Het resultaat is weergegeven in meter, in Tabel 4.1. Hierbij zien we dat het maximum verschil 0.5214 meter bedraagt indien de persoon zichtbaar is voor drie camera's, en 1.0586 meter indien de persoon zichtbaar is voor twee camera's. Het maximum voor drie camera's lijkt echter een uitschieter aangezien het gemiddelde verschil 0.2914 meter bedraagt. De grootste fout doet zich voor wanneer de persoon zichtbaar is voor twee camera's die bijna recht tegenover elkaar staan. Een kleine verschuiving van één van de projectielijnen zorgt er namelijk voor dat het snijpunt met de andere projectielijn een grote verplaatsing ondergaat.

Tabel 4.1 Eigen data (15 frames per reeks): verschil met werkelijke positie

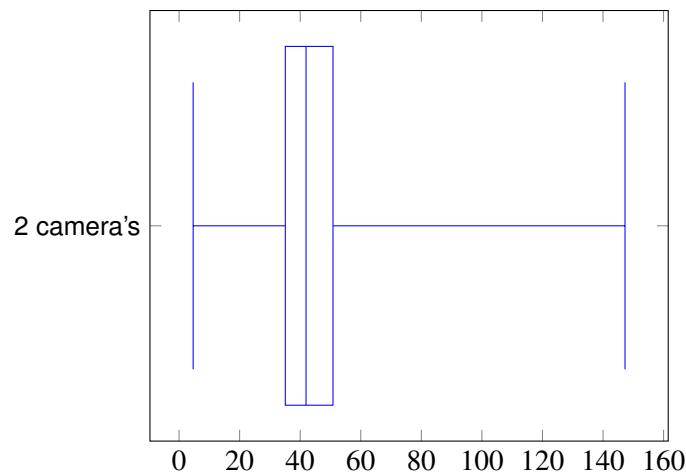
	Mimimum[m]	Maximum[m]	Gemddelde[m]
Drie camera's	0.1104	0.5214	0.31202
Twee camera's	0.0423	1.0586	0.2914

4.2 CMU-dataset

Aangezien het niet voldoende nauwkeurig is om onze implementatie enkel te valideren op eigen data, hebben we de resultaten ook gevalideerd op de CMU-dataset (Joo et al. (2015)). Hiervoor hebben we de 150821_dance4 data gebruikt. Deze data bevatten de 3D-positie van één persoon die beweegt gedurende een aantal minuten. Om te valideren hebben we echter ook de 2D-pose van de persoon voor verschillende camera's nodig. Om dit te bekomen hebben we twee opties. Een eerste mogelijkheid is het uitvoeren van OpenPose op de data. Het nadeel van deze optie is echter dat we hierdoor niet alleen de nauwkeurigheid van onze implementatie in rekening brengen, maar ook de nauwkeurigheid van OpenPose. Dit zou een verkeerd beeld geven van de nauwkeurigheid van ons eigen. Een tweede mogelijkheid bestaat erin om de CMU-dataset te gebruiken. De CMU-dataset bevat namelijk de 3D-positie van de personen in beeld. Deze 3D-positie bestaat uit een top-down positie (wat wij willen berekenen) en een 3D-pose. Aangezien de kalibratieparameters gekend zijn voor alle camera's die gebruikt werden bij het creëren van de CMU-dataset, is het mogelijk om een projectie uit te voeren om op deze manier de 2D-pose voor elke camera te berekenen. Aangezien we op deze manier de nauwkeurigheid van OpenPose niet mee in rekening nemen, en de pose estimator in de toekomst makkelijk vervangen kan worden door een andere, kiezen we voor deze optie.

De implementatie om deze projectie uit te voeren werd beschikbaar gesteld door de makers van de CMU-dataset. Na enkele kleine aanpassingen worden de bekomen 2D-posities opgeslagen onder de vorm van JSON-files, waardoor we deze gegevens kunnen gebruiken voor onze eigen implementatie om de top-down positie te bepalen. Door tijdsgebrek was het echter niet mogelijk om de kalibratieparameters te gebruiken voor het plaatsen van de camera's op een plattegrond van de Panoptic Studio. We hebben voor een 2100-tal frames de werkelijke positie en de door ons algoritme benaderde positie berekend. Het verschil tussen beide wordt weergegeven in Figuur 4.1.

De gemiddelde afwijking op de beelden uit de CMU-dataset ligt iets hoger (0.4763 meter) dan bij onze eigen data. Ook het grootste verschil met de werkelijke positie (147.262 mm) is iets groter dan bij de eigen data. Dit kunnen we mogelijk verklaren doordat het manueel positioneren van de verschillende camera's voor deze beelden niet makkelijk was, aangezien er geen voorwerpen in de panoptic-studio aanwezig zijn. Dit maakt het moeilijker om puntenkoppels te vormen tussen de



Figuur 4.1: Boxplot van het verschil tussen werkelijke positie en de door onze techniek benaderde positie uitgedrukt in millimeter (2100 frames)

camerabeelden en de plattegrond. Het minimum verschil tussen de werkelijke en benaderde positie is ongeveer gelijk aan het minimum verschil bij onze eigen data. Het eerste en derde kwartiel liggen respectievelijk op 35.0836m en 50.8325m. Met een gemiddelde van 0.4695 meter tonen we aan dat onze techniek gebruikt kan worden voor het benaderen van de top-down positie van een persoon in een leefruimte. Ons doel was immers om de positie van de persoon voldoende nauwkeurig te bepalen zodat een PTZ camera hem of haar zelfstandig in beeld zou kunnen brengen. Ook wanneer de camera zich daarbij richt op een punt dat in werkelijkheid een kleine halve meter van de persoon verwijderd is, zal deze nog steeds in beeld zijn.

Hoofdstuk 5

Conclusie

Het doel van dit onderzoek was het benaderen van de positie van een persoon in een leefruimte. Om de 3D-pose van de persoon te bepalen hebben we in eerste instantie VNect onderzocht. Deze techniek werkt echter niet bij occlusie, waardoor we besloten om ons te focussen op de top-down positie. We kozen ervoor om het OpenPose framework te gebruiken om de 2D-pose van de persoon te benaderen op afbeeldingen, omdat dit framework meer informatie biedt over de pose van de persoon dan gewone detectietechnieken. Vervolgens hebben we een aantal kalibratietechnieken bekeken, waarbij het five-point pose algoritme ons het meest geschikt leek om te gebruiken in een leefruimte. Doordat dit algoritme niet meteen werkte zoals het hoorde, hebben we ervoor gekozen om de camera's handmatig te kalibreren. Uit het onderzoek is gebleken dat we de top-down positie van een persoon kunnen benaderen door gebruik te maken van de hoek-per-pixel methode. Hierbij worden er projectielijnen gevormd op de plattegrond van de leefruimte, tussen de camera en de persoon. De positie van de persoon is dan bij benadering gelijk aan het snijpunt van deze projectielijnen. We hebben onze implementatie geëvalueerd op eigen data en op data van de CMU-dataset (Joo et al. (2015)). De positie die we benaderen op de CMU-dataset heeft een gemiddelde afwijking van 0.4763 meter ten opzichte van de effectieve positie, en dit voor een 2100-tal frames.

5.1 Toekomst

In de toekomst is er nog verbetering mogelijk. Zo zouden we de grootte van de persoon kunnen gebruiken om de positie nauwkeuriger te benaderen indien de persoon slechts zichtbaar is voor één camera. Ook zou het gebruik van nauwkeurigere pose estimation technieken het resultaat op de eigen data kunnen verbeteren. De handmatige kalibratie zou vervangen kunnen worden door een kalibratietechniek die het mogelijk maakt om de externe kalibratieparameters te bepalen voor camera's die ver van elkaar verwijderd zijn. Verder zou het mogelijk gemaakt kunnen worden om de posities van meerdere mensen te benaderen en om aanpassingen te maken voor camera's die hoger gepositioneerd zijn.

Bibliografie

- Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723.
- Amin, S., Andriluka, M., Rohrbach, M., and Schiele, B. (2013). Multi-view pictorial structures for 3d human pose estimation. pages 45.1–45.11.
- Amit, Y. and Felzenszwalb, P. (2014). Object detection.
- Andriluka, M., Pishchulin, L., Gehler, P., and Schiele, B. (2014). 2d human pose estimation: New benchmark and state of the art analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Bashari Rad, B., Bhatti, H., and Ahmadi, M. (2017). An introduction to docker and analysis of its performance. 173:8.
- Beardsley, P. and Murray, D. (1992). *Camera Calibration using Vanishing Points*, pages 416–425. Springer London, London.
- Cao, Z., Simon, T., Wei, S., and Sheikh, Y. (2017). Realtime multi-person 2d pose estimation using part affinity fields. pages 1302–1310.
- Chen, C. and Ramanan, D. (2016). 3d human pose estimation = 2d pose estimation + matching. *CoRR*, abs/1612.06524.
- Chen, Y., Shen, C., Wei, X., Liu, L., and Yang, J. (2017). Adversarial posenet: A structure-aware convolutional network for human pose estimation. *CoRR*, abs/1705.00389.
- Chetlur, S., Woolley, C., Vandermersch, P., Cohen, J., Tran, J., Catanzaro, B., and Shelhamer, E. (2014). cudnn: Efficient primitives for deep learning. *CoRR*, abs/1410.0759.
- Dalal, N. and Triggs, B. (2005). Histogram of oriented gradients for human detection. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Diego, CA, USA, USA, June 20–25, 2005, CVPR’05*. IEEE.

- Dollar, P., Tu, Z., Perona, P., and Belongie, S. (2009). Integral channel features. In *Proc. BMVC*, pages 91.1–91.11. doi:10.5244/C.23.91.
- Ershadi-Nasab, S., Noury, E., Kasai, S., and Sanaei, E. (2017). Multiple human 3d pose estimation from multiview images. Technical report, Sharif University of Technology, Iran.
- Fang, H., Xie, S., and Lu, C. (2016). RMPE: regional multi-person pose estimation. *CoRR*, abs/1612.00137.
- Felzenszwalb, P. F., Girshick, R. B., McAllester, D., and Ramanan, D. (2010). Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645.
- Fischler, M. A. and Elschlager, R. A. (1973). The representation and matching of pictorial structures. *IEEE Trans. Comput.*, 22(1):67–92.
- Fleuret, F., Berclaz, J., Lengagne, R., and Fua, P. (2008). Multi-camera people tracking with a probabilistic occupancy map. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):267–282.
- Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *In Proc. of Fourth Alvey Vision Conference*, pages 147–151.
- Hartley, R. I. and Zisserman, A. (2004). *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition.
- He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition. *CoRR*, abs/1512.03385.
- Heikkila, J. and Silven, O. (1997). A four-step camera calibration procedure with implicit image correction. Technical report, Infotech Oulu and Department of Electrical Engineering University of Oulu.
- Hödlmoser, M. and Kampel, M. (2010). *Multiple Camera Self-calibration and 3D Reconstruction Using Pedestrians*, pages 1–10. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Hofmann, M. and Gavrila, D. M. (2012). Multi-view 3d human pose estimation in complex environment. *International Journal of Computer Vision*, 96(1):103–124.
- Hombali, A., Gorde, V., and Deshpande, A. (2011). Monocular depth perception using image processing and reinforcement learning. In *International Conference on Graphic and Image Processing (ICGIP 2011)*, volume 8285, page 82850J.

- Huang, D., Bevilacqua, V., and Premaratne, P., editors (2014). *Intelligent Computing Theory - 10th International Conference, ICIC 2014, Taiyuan, China, August 3-6, 2014. Proceedings*, volume 8588 of *Lecture Notes in Computer Science*. Springer.
- Insafutdinov, E., Pishchulin, L., Andres, B., Andriluka, M., and Schiele, B. (2016). Deepcut: A deeper, stronger, and faster multi-person pose estimation model. volume abs/1605.03170.
- Ionescu, C., Papava, D., Olaru, V., and Sminchisescu, C. (2014). Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339.
- Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R. B., Guadarrama, S., and Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. *CoRR*, abs/1408.5093.
- Johnson, S. and Everingham, M. (2010). Clustered pose and nonlinear appearance models for human pose estimation. In *Proceedings of the British Machine Vision Conference*. doi:10.5244/C.24.12.
- Joo, H., Liu, H., Tan, L., Gui, L., Nabbe, B., Matthews, I., Kanade, T., Nobuhara, S., and Sheikh, Y. (2015). Panoptic studio: A massively multiview system for social motion capture. In *The IEEE International Conference on Computer Vision (ICCV)*.
- Li, S. and Chan, A. B. (2014). 3d human pose estimation from monocular images with deep convolutional neural network. In *Computer Vision - ACCV 2014 - 12th Asian Conference on Computer Vision, Singapore, Singapore, November 1-5, 2014, Revised Selected Papers, Part II*, pages 332–347.
- Lin, T., Maire, M., Belongie, S. J., Bourdev, L. D., Girshick, R. B., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312.
- Liu, C., Hu, Y., Li, Y., Song, S., and Liu, J. (2017). PKU-MMD: A large scale benchmark for continuous multi-modal human action understanding. *CoRR*, abs/1703.07475.
- Mehta, D., Rhodin, H., Casas, D., Sotnychenko, O., Xu, W., and Theobalt, C. (2016). Monocular 3d human pose estimation using transfer learning and improved CNN supervision. *CoRR*, abs/1611.09813.
- Mehta, D., Sridhar, S., Sotnychenko, O., Rhodin, H., Shafiei, M., Seidel, H., Xu, W., Casas, D., and Theobalt, C. (2017). Vnect: Real-time 3d human pose estimation with a single RGB camera. *CoRR*, abs/1705.01583.

- Moon, T. K. (1996). The expectation-maximization algorithm. *IEEE Signal Processing Magazine*, 13(6):47–60.
- Newell, A. and Deng, J. (2016). Associative embedding: End-to-end learning for joint detection and grouping. *CoRR*, abs/1611.05424.
- Nisér, D. (2004). An efficient solution to the five-point relative pose problem. In *Transactions on Pattern Analysis and Machine Intelligence*, pages 756–770. IEEE.
- Pavlakos, G., Zhou, X., Derpanis, K. G., and Daniilidis, K. (2016). Coarse-to-fine volumetric prediction for single-image 3d human pose. *CoRR*, abs/1611.07828.
- Pavlakos, G., Zhou, X., Derpanis, K. G., and Daniilidis, K. (2017). Harvesting multiple views for marker-less 3d human pose annotations. *CoRR*, abs/1704.04793.
- Pishchulin, L., Insafutdinov, E., Tang, S., Andres, B., Andriluka, M., Gehler, P. V., and Schiele, B. (2015). Deepcut: Joint subset partition and labeling for multi person pose estimation. volume abs/1511.06645.
- Rasmussen, C. E. (2000). The infinite gaussian mixture model. In *In Advances in Neural Information Processing Systems 12*, pages 554–560. MIT Press.
- Redmon, J. and Farhadi, A. (2016). YOLO9000: better, faster, stronger. *CoRR*, abs/1612.08242.
- Schiele, B. (2012). A database for fine grained activity detection of cooking activities. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR ’12, pages 1194–1201, Washington, DC, USA. IEEE Computer Society.
- Shahroudy, A., Liu, J., Ng, T., and Wang, G. (2016). NTU RGB+D: A large scale dataset for 3d human activity analysis. *CoRR*, abs/1604.02808.
- Sigal, L. (2014). *Computer Vision*. Springer US.
- Simon, T., Joo, H., Matthews, I. A., and Sheikh, Y. (2017). Hand keypoint detection in single images using multiview bootstrapping. *CoRR*, abs/1704.07809.
- Sutton, C. and McCallum, A. (2012). An introduction to conditional random fields. *Found. Trends Mach. Learn.*, 4(4):267–373.
- Tekin, B., Katircioglu, I., Salzmann, M., Lepetit, V., and Fua, P. (2016). Structured prediction of 3d human pose with deep neural networks. *CoRR*, abs/1605.05180.
- Tekin, B., Rozantsev, A., Lepetit, V., and Fua, P. (2015). Direct prediction of 3d body poses from motion compensated sequences. *CoRR*, abs/1511.06692.

- Viola, P., Jones, M. J., and Snow, D. (2003). Detecting pedestrians using patterns of motion and appearance. In *Proceedings of the Ninth IEEE International Conference on Computer Vision, Nice, France, France, October 13–16, 2003, ICCV '03*. IEEE.
- Wang, C., Wang, Y., Lin, Z., Yuille, A. L., and Gao, W. (2014). Robust estimation of 3d human poses from a single image. *CoRR*, abs/1406.2282.
- Wei, S., Ramakrishna, V., Kanade, T., and Sheikh, Y. (2016a). Convolutional pose machines. volume abs/1602.00134.
- Wei, S., Ramakrishna, V., Kanade, T., and Sheikh, Y. (2016b). Convolutional pose machines. *CoRR*, abs/1602.00134.
- Yang, B., Yan, J., Lei, Z., and Li, S. Z. (2014). Aggregate channel features for multi-view face detection. In *IEEE International Joint Conference on Biometrics*, pages 1–8.
- Yasin, H., Iqbal, U., Krüger, B., Weber, A., and Gall, J. (2015). 3d pose estimation from a single monocular image. *CoRR*, abs/1509.06720.
- Zhou, X., Zhu, M., Leonardos, S., Derpanis, K. G., and Daniilidis, K. (2015). Sparseness meets deepness: 3d human pose estimation from monocular video. *CoRR*, abs/1511.09439.

FACULTEIT INDUSTRIELE INGENIEURSWETENSCHAPPEN
TECHNOLOGIECAMPUS DE NAYER
Jan De Nayerlaan 5
2860 SINT-KATELIJNE-WAVER, België
tel. + 32 15 31 69 44
iiw.denayer@kuleuven.be
www.iiw.kuleuven.be



LID VAN
**ASSOCIATIE
KU LEUVEN**