Inhoudsopgave

Hoofdstuk 1

Inleiding

Menselijke actieherkenning is het proces dat op een automatische manier (I) detecteert dat een persoon een bepaalde actie uitvoert en (II) herkennen welke actie dit is. Voorbeelden van zulke acties zijn lopen, stappen, zwaaien, springen, bukken, enz. Actieherkenning kent tal van toepassingen, voornamelijk bij de interactie tussen mens en computer, zoals fysiotherapie [1], lichaamsanalyse [2] en _ToDo: nog wat voorbeelden zoeken

Actieherkenning wordt voornamelijk gerealiseerd met kleuren- en dieptebeelden.

1.1 Probleemstelling

In het onderzoeksgebied actieherkenning is er al uitbundig onderzoek gedaan. Zo is men in staat om voor zowel kleurenbeelden als dieptebeelden

In de literatuur wordt er vaak een speciale pose, de startpose, verwacht [3], [1]. Dit is de pose die een persoon moet aannemen vooraleer actieherkenning uitgevoerd zal worden.

Actieherkenning en actiedetectie blijft moeilijk te realiseren in een real-time scenario, waarbij snelle beslissingen gemaakt moeten worden. Het doel van deze masterproef is om eerst te onderzoeken welke methoden er al bestaan op vlak van actieherkenning en hoe deze efficiënter kunnen gemaakt worden. De algemene aanpak wordt beschreven in hoofdstuk 2.

1.2 De Kinect

De Kinect (figuur 1.1) is origineel ontworpen als manier om de gebruiker zelf als spelcontroller te beschouwen. Voor deze masterproef wordt de tweede versie van de Kinect gebruikt (figuur 1.1b) en wordt kort besproken.

De Kinect bevat meerdere sensoren waaronder een kleurencamera, dieptecamera en infraroodcamera. Op basis van de dieptebeelden is de Kinect in staat om een skelet te genereren voor maximaal zes personen, elk met 25 joints die belangrijke kenmerken van het menselijk lichaam voorstellen, meestal op de plaats van gewrichten.

(a) De originele Kinect sensor, ontwikkeld int 2010 voor de Xbox 360.

(b) De tweede iteratie van de Kinect sensor, specifiek gemaakt voor Xbox One en uitgebracht in 2013.

Figuur 1.1: Twee versies van de Kinect sensor.

1.3 Structuur

In hoofdstuk 2 wordt de algemene methodologie beschreven die gebruikt wordt om deze masterproef te realiseren, waarom deze gebruikt wordt en wat het beoogde eindresultaat is. Verder wordt er in hoofdstuk **??** een overzicht gegeven van bestaande methoden en welke features en classifiers zij gebruiken.

Hoofdstuk 2

Methodologie

Ieder persoon heeft een eigen interpretatie van een bepaalde actie. Er kunnen verschillen zijn in snelheid, de positie relatief tot het hele lichaam en zelfs de lichaamsbouw van die persoon. Elk van deze variaties in een algoritme steken is dan ook onbegonnen werk. Daarom maakt elk actieherkenningsalgoritme op één of andere manier gebruik van machine learning. Op basis van training data wordt een classifier getraind die kan voorspellen tot welke klasse een nieuwe observatie behoort. Observaties bij een Kinect zijn voornamelijk kleurenbeelden of dieptebeelden.

Zulke observaties worden getransformeerd naar *features*. Dit zijn meetbare eigenschappen of karakteristieken van het object dat geobserveerd wordt. Deze eigenschappen moeten bovendien voldoende onderscheidend zijn zodat het mogelijk is om de observatie te klasseren. Een feature tracht de originele observatie te reduceren tot bruikbare informatie om op een eenvoudigere manier classificatie uit te voeren. Een *feature vector* vormt een n-dimensionale wiskundige vector van features. Elke dimensie van deze vector is een individuele feature en vormt de *feature space*. Via bestaande features kunnen er nieuwe features aangemaakt worden via *feature construction*. Het proces om een observatie om te vormen tot een feature vector wordt gerealiseerd met een *feature descriptor*.

Een *classifier* verwacht als input zo een feature vector. Het is de taak van een classifier om te bepalen tot welke klasse een nieuwe observatie behoort. In het geval van actieherkenning is de klasse een bepaalde actie, zoals zwaaien, bukken of springen. Een classifier zal bij het bepalen van een klasse ook een zogenaamde *score* geven. Dit is de waarschijnlijkheid dat de voorspelde klasse correct is. Een eenvoudige *lineaire classifier* berekent de score op basis van een lineaire combinatie door het kruisproduct te nemen tussen de feature vector en een speciale gewichtenvector, specifiek voor die klasse en gebaseerd op de training data. De voorspelde klasse is dan die met de hoogste score. Er bestaan zowel *supervised* als *unsupervised* classificatiemodellen. Beiden maken gebruik van een leerverzameling; een collectie van voorbeelden waaruit het systeem moet leren. Voor een supervised model wordt het gewenste resultaat meegegeven aan elk object in de leerverzameling. Bij een unsupervised model is dit niet zo, maar er is wel een algemeen idee van wat er moet aangeleerd worden.

Een supervised model wordt vaak toegepast op actieherkenning: de leerverzameling bevat videobeelden waarin acties door personen worden uitgevoerd, samen met de gelabelde klasse. Voor actieherkenning speelt het tijdelijke aspect een grote rol. Het is daarom dan ook minder interessant om slechts voor één frame de juiste actie te classificeren. Wel interessant is om voor een

verzameling van frames na te gaan welke actie deze frames voorstellen.

_ToDo: eigen inbreng vanaf nu

- Basisidee: *key frames* = kan zeker voordeel brengen.
  - Welke acties herkennen?
  - Wat is 'te weinig verschil'? Zie bv [4].
  - Wanneer gebruikte frames weggooien? Ik beslis welke frames niet opgenomen worden, dus er is vrij veel bias.
  - Studie hidden markov model → zie 5.2.3
  - Waarom niet gewoon simpele teller gebruiken om met tijdsaspect om te gaan?
- Waarom is dit onderzoek nuttig?
  - Live actieherkenning vereist snelle classificatie
  - Verlagen van computationele kost
  - Op voorwaarde dat bepalen van keyframes sneller is dan gewoon elke frame in beschouwing te nemen.
  - Wat is live actieherkenning?
    - Er is geen default pose
    - Er is niet altijd een actor in beeld. Een actor is een persoon waarvan de skeletinformatie beschikbaar is, dus als de kinect correct het skelet kan bepalen van een persoon.
    - Vanaf dat een actor een actie uitvoert, moet deze vroeg genoeg herkend kunnen worden (< 1 seconde, liefst sneller)
    - De classificatie moet ook kunnen omgaan met het tijdsaspect van de uitgevoerde actie.
- Waarom skeletbeelden van de Kinect gebruiken?
  - Worden gegenereerd uit dieptebeelden, op een vrij efficiënte manier [5].
  - Dieptebeelden zijn ongevoelig voor verandering van lichtintensiteit. Ook vormt schaduw geen probleem meer.
  - Nadelen:
    - De kinect mag het enige apparaat in de omgeving zijn die infrarood uitstraalt. Dus kan al enkel indoor gebruikt worden.
    - mogelijke fouten door ruis in dieptebeelden
  - Kenmerken:
    - Verzameling van joints.
    - Elke joint wordt voorgesteld door een $3D$ coördinaat.

- Classificatiemodel pas vastleggen nadat verschillende mogelijkheden getest zijn op dataset.
  - Support vector machines
  - ensemble methoden
  - _Opmerking: nog geen prioriteit

2.1 Implementatie

Om enigszins het aantal geschreven code tot een minimum te houden wordt er gekozen om Python te gebruiken in combinatie met scikit-learn.

(voor mezelf, dingen die ik zeker nog op laptop moet uitvoeren:)

- `python -m pip install --upgrade pip`
- `python -m pip install -U matplotlib`
- `pip install -U scikit-learn`
- Zorgen dat tkinter gecheckt is bij installatie (eventueel installatie opnieuw uitvoeren met MODIFY)

Hoofdstuk 3

Features

In dit hoofdstuk wordt bestaande literatuur voor actieherkenning verkent en besproken. De voornaamste methoden maken enerzijds gebruik van kleurenbeelden [6], [7], [8], [9] en anderzijds van dieptebeelden [10], [11], [12], [13]. Onafhankelijk van welke beelden die gekozen worden, moeten er features geëxtraheerd worden. Volgende sectie bespreekt kort de voornaamste manieren om features te bepalen uit zowel kleuren- als dieptebeelden.

3.1 Features uit kleurenbeelden

In het geval van kleurenbeelden kan het bepalen van features in twee categorieën onderverdeeld worden: *globale feature extraction* en *lokale feature extraction* [14]. Bij een globale aanpak wordt een persoon eerst gelokaliseerd met behulp van background subtraction of tracking gevolgd door het encoderen van de interesseregio's. Deze representatie kan veel informatie bevatten, maar door de nood aan background subtraction of tracking is deze methode gevoelig aan ruis.

Een lokale aanpak tracht deze problemen te vermijden door eerst lokale interessepunten, in de literatuur Spatio-Temporal Interest Points (STIP) genoemd, te bepalen. Deze interessepunten kunnen zowel het tijdelijke als het ruimtelijke aspect modelleren. Rond deze interessepunten worden patches berekent, afhankelijk van gekozen parameters. De verzameling van zulke patches is de representatie. Deze methode geniet de voorkeur omdat background subtraction of tracking niet strikt noodzakelijk is zodat het minder gevoelig is aan ruis.

feature detectors: _ToDo: harris3D [6] _ToDo: cuboid [7] _ToDo: Hessian [8] _ToDo: dense trajectory [9]

feature descriptors: _ToDo: cuboid descriptor: [7] _ToDo: hog/hof [6]

3.2 Features uit dieptebeelden

De methoden die bruikbaar zijn op kleurenbeelden zijn niet bruikbaar op dieptebeelden. Recente literatuur over feature extraction op dieptebeelden levert vaak algoritmen [12], [15], [16] op die gebruik maken van skeletbeelden gegeneerd met methode [5] waarop de skelettracker van de Kinect op gebaseerd is. De skeletbeelden van de Kinect geven de drie-dimensionale coördinaten van 25 punten, *joints* genoemd, die belangrijke kenmerken van het menselijk lichaam voorstellen

_ToDo: figuur van de joints . Zo een skelettracker neemt de taak van een feature detector op zich, zodat algoritmen die gebruik maken van skeletbeelden enkel feature descriptors moeten implementeren. Deze joints bevatten echter geen tijdelijke informatie zodat dit op een andere manier moet gemodelleerd worden. Vaak gebruikte modellen zijn Hidden Markov Model (HMM) ([12], [13]), Conditional Random Field (CRF) ([17]), Dynamic Temporal Warping (DTW) ([18]) en Fourier Temporal Pyramid (FTP) ([15]). Al deze modellen zijn temporale classifiers en worden kort besproken in sectie 4.2.

Het gebruik van de Kinect is geen vereiste om actieherkenning met dieptebeelden uit te voeren. [10] stelt elke frame voor als een verzameling van 3D punten, geëxtraheerd uit de silhouetten dat de dieptebeelden geven en maken gebruik van een HMM om het tijdelijke aspect te modelleren. [11]

In volgende onderdelen worden een aantal belangrijke feature descriptors beschreven.

3.2.1 Histograms of 3D Joints (HOJ3D)

Het werk van [12] toont aan hoe een histogram van drie-dimensionale punten aan real-time actieherkenning kan doen. Ze transformeren het skeletbeeld gegenereerd door de Kinect om in bolcoördinaten. Als oorsprong wordt de heup joint genomen. De horizontale referentievector $\vec{\alpha}$ wordt parallel met de grond genomen door de oorsprong. De verticale referentievector $\vec{\theta}$ staat loodrecht op $\vec{\alpha}$ en gaat ook door de oorsprong. De drie-dimensionale ruimte wordt opgesplitst in 84 deelruimten. Elke joint zal zich dan in één van die 84 deelruimte bevinden. Een histogram wordt opgemaakt door elke joint te wegen in 8 naburige deelruimten via een Gaussische functie:

$$p(X, \mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(X-\mu)^T \Sigma^{-1}(X-\mu)}$$

Via Linear Discriminant Analysis (LDA) wordt uit het bekomen histogram features geëxtraheerd.

Hoofdstuk 4

Classificatie

Op het moment dat een feature vector opgebouwd is voor een individuele frame of een verzameling van frames is het actieherkenningprobleem gereduceerd tot een classificatieprobleem. De bekomen feature vector wordt als input aan een classifier gegeven die dan tracht de juiste klasse toe te kennen op basis van de leerverzameling. Voor actieherkenning kunnen drie algemene methoden beschreven worden:

1. Classificeren zonder de tijdelijke dimensie expliciet te modelleren, *directe classificatie*;
2. Classificeren waarbij de tijdelijke dimensie wel gemodelleerd wordt, *temporale modellen*;
3. Algemene classificatie zonder de actie te modelleren, *actiedetectie*.

4.1 Directe classificatie

Wanneer het tijdelijke aspect verworpen wordt zijn er maar twee mogelijkheden: alle geobserveerde frames vervatten in één enkele representatie of actieherkenning uitvoeren op elke individuele frame. Een voorbeeld van zo een algoritme is k-nearest neighbours. Deze methode maakt gebruik van de afstand van de geobserveerde feature vector tot elke andere feature vector uit de leerverzameling. Uit de k dichtste buren wordt dan de klasse genomen die het meest voorkomt. Deze operatie vraagt voor een grote leerverzameling veel rekenkost omdat elke feature vector gecontroleerd moet worden. Deze methode kan zowel op frame-niveau als op sequentieniveau werken. In het tweede geval wordt er met bijvoorbeeld majority voting de klasse gekozen die het meest voorkomt in een sequentie van frames.

4.2 Temporale modellen

Bij temporale modellen zijn er twee grote klassen te onderscheiden: *generatieve* en *discriminerende* modellen. Een generatief algoritme modelleert hoe een input x wordt gegenereerd om zo een actieklasse y toe te kennen en maakt gebruik van de gezamenlijke kansverdeling $P(x, y) = P(x|y)P(y)$. Een generatief model zal dus de kansverdeling van de leerverzameling modelleren en zal bij een nieuwe observatie. Een discriminerend model zal de kans $P(y|x)$ direct modelleren zodat een directe mapping van x op y beschikbaar is. De onderliggende kansverdelingen worden dan ook niet berekent.

Er bestaat een grote discussie over welk van deze twee modellen nu de voorkeur krijgen. Er is aangetoond [19] dat discriminerende modellen de voorkeur genieten.

4.2.1 Generatieve modellen

Een HMM is een typisch voorbeeld van een generatief model. Een HMM is een uitbreiding van een Markov model waarbij de staat van elke toestand nu verborgen is. De toestanden zijn hier de verschillende fasen van een bepaalde actie. Een HMM gaat uit van twee assumpties:

1. Een verandering van toestand hangt enkel af van de vorige toestand. Dit wordt ook de *Markov eigenschap* genoemd.
2. De observatie behorend bij een toestand is onafhankelijk van elke andere observatie.

Voor een verzameling met n klassen $\Lambda = \lambda^1 ... \lambda^n$ en een verzameling met k observaties $O = \{o_1 ... o_k\}$, moet de juiste klasse λ geselecteerd worden die de kans op $P(\lambda|O)$ maximaliseert.

$$\lambda = \arg \max_{1 \leq i \leq n} P(O|\lambda^i)$$

4.2.2 Discriminerende modellen

Conditional Random Fields

Een HMM gaat ervan uit de observaties onafhankelijk zijn van elkaar, wat niet altijd het geval is. Een CRF is een voorbeeld van een discriminerend model. Een discriminerende classifier houdt rekening met meerdere observaties in de tijd en is geschikt voor de classificatie van een reeks van observaties. Een CRF gaat ervan uit dat de volgorde van observaties wel degelijk een impact heeft op de betekenis van deze observaties.

Dynamic Time Warping

DTW is een algoritme dat de gelijkenis tussen twee sequenties bestudeerd, die verschillend kunnen zijn in snelheid. In actieherkenning lost dit het probleem van verschillen in actiesnelheid op. Een persoon die trager of sneller zwaait dan een persoon in de leerverzameling, kan via DTW toch herkent worden. Veronderstel twee verzamelingen van feature vectoren $X = \{x_1, x_2, ...x_N\}$ en $Y = \{y_1, y_2, ..., y_M\}$, een kostfunctie $c(x, y)$ die de kost bepaald tussen twee feature vectoren en $p = \{p_1, ..., p_L\}$ met $p_l = (n_l, m_l)$

Een DTW heeft bepaalde restricties:

- $p_1 = (1, 1)$ en $p_L = (N, M)$.
- $n_1 \leq n_2 \leq ... \leq n_L$ en $m_1 \leq m_2 \leq ... \leq m_L$.

4.3 Key frames

In plaats van elke frame te classificeren, zou een actie kunnen voorgesteld worden door *key frames*. Dit is een selectie van frames die een actie voldoende kunnen voorstellen zodanig dat verschillende acties nog steeds onderscheidbaar zijn en variaties van dezelfde actie hetzelfde gelabeld worden. In de literatuur bestaan er diverse manieren om zulke key frames te bepalen.

De methode van [4] berekent een verschil op basis van de joints die de Kinect beschikbaar stelt. De methode van [20] maakt gebruik van randdetectie om silhouette te bekommen en zoekt een match tussen voorgedefinieerde silhouetten die een key frame van een actie voorstelt. Andere methoden maken ook gebruik van voorgedefinieerde exemplaren [21], [22],

Hoofdstuk 5

notities

5.1 papers

- Bron [23]
  - Actieherkenning en actiedetectie systeem voor temporally untrimmed videos door de combinatie van motion en appearance features.
    - Motion feature = Fisher vector representatie met dense trajectories
    - Appearance feature: deep convolutional neural network
- Bron [24]
  - Actieherkenning = het herkennen van een actie binnen een goed gedefinieerde omgeving
  - Actiedetectie = het herkennen en lokaliseren van acties(begin, duratie en einde) in de ruimte en de tijd
  - training set = wordt gebruikt om classifier te trainen
  - validation set = optioneel, bevat andere data dan de training set om de classifier te optimaliseren
  - testing set = testen van de classifier (performance)
  - Drie manieren om dataset op te splitsen in deze drie sets:
    - voorgedefinieerde split: De dataset wordt opgesplitst in twee of drie delen zoals de auteurs van die dataset dat vermelden
    - n-voudige cross-validatie: Verdeeld de dataset in n gelijkvoudige stukken. Hierbij worden er (n-1)/n percentage van de videos gebruikt om te trainen, en dan de overige 1/n om te testen. Dit proces wordt n keer herhaald, zodat elke video éénmaal gebruikt werd voor te testen
    - leave-one-out cross-validatie: aangewezen methode indien testdata personen zijn. 1 persoon wordt als testset beschouwd. De overige $n-1$ personen is de training set.

  - om actieklasse te bepalen = features extraheren en in classifier steken → classifier bepaalt actieklasse
  - Temporally untrimmed video = delen van de video bevatten GEEN ENKELE actie. Variaties van dezelfde actie kan op hetzelfde moment voorkomen
  - THUMOS challenge:
  - 2015 → slechts één team heeft detection challenge geprobeerd
  - Classificatietaak: de lijst van acties geven die in een lange, niet getrimde video voorkomen
  - Detectietaak: ook de lijst van acties geven PLUS de plaats in tijd waar ze voorkomen
- Bron [5] gaat eerder over hoe het skelet bepaalt wordt
  - voorstel van een methode om op een accurate manier de 3D posities van de joints te bepalen, vanuit slechts één dieptebeeld, zonder temporale informatie
  - Het bepalen van lichaamsdelen is invariant van pose, lichaamsbouw, kleren, etc...
  - Kan runnen aan 200 fps
  - Wordt effectief gebruikt in de Kinect software (onderzoeksteam is van Microsoft)
  - Een dieptebeeld wordt gesegmenteerd in verschillende lichaamsdelen, aangegeven door een kleur, op basis van een kansfunctie; Elke pixel van het lichaam wordt apart behandeld en gekleurd. Een verzameling van dezelfde kleuren wordt een joint
  - Aangezien tijdsaspect weg is, is er enkel interesse in de statische poses van een frame. Verschillen van pose in twee opeenvolgende frames is miniscuul zodat die genegeerd worden
- Bron [10] (pre-kinect era)
  - Actieherkenning met behulp van reeksen van dieptebeelden
  - Gaan ervan uit dat efficiënte tracking van skeletbeelden nog niet mogelijk is. (is gepubliceerd zelfde jaar dat Kinect beschikbaar was, 2010)
  - Hun oplossing is dus niet gebaseerd op het tracken van de skeletbeelden
- Bron [25]
  - Probleem: output van de actiecategorie EN de start en eind tijd van de actie.
  - Ze beweren dat actieherkenning reeds goed opgelost is, maar niet actiedetectie. Hun definities zijn:
    * Actieherkenning: De effectieve actieherkenning indien het systeem weet wanneer hij moet herkennen
    * Actiedetectie: een langdurige video, waarbij de start en stop van een actie niet gedefinieerd zijn = untrimmed video (videos waarbij er meerdere acties op hetzelfde moment kunnen voorkomen, alsook een irrelevante achtergrond). sluit heel goed aan op onze masterproef

- Uitdaging in bestaande oplossingen: groot aantal onvolledige actiefragmenten. Voorbeelden:
  - Bron [26]:
    - maakt gebruik van **untrimmed classificatie**: de top $k = 3$ (bepaalt via cross-validation) labels worden voorspelt door globale video-level features. Daarna worden frame-level binaire classifiers gecombineerd met dynamisch programmeren om de activity proposals (die getrimmed zijn) te genereren. Elke proposal krijgt een label, gebaseerd op de globale label.
  - Bron [27]:
    - Spreekt over de onzekerheid van het voorkomen van een actie en de moeilijkheid van het gebruik van de continue informatie
    - Pyramid of Score Distribution Feature (PSDF) om informatie op meerdere resoluties op te vangen
    - PSDF in combinatie met Recurrent Neural networks bieden performantiewinst in untrimmed videos.
    - Onbekende parameters: actielabel, actieuitvoering, actiepositie, actielengte
    - Oplossing? Per frame een verzameling van actielabels toekennen, gebruik makend van huidige frame actie-informatie en inter-frame consistentie = PSDF
- De moeilijkheid is: start, einde en duur van de actie te bepalen.
- Hun oplossing is **Structured Segment Network**:
  - input: video
  - output: actiecategorieën en de tijd wanneer deze voorkomen
  - Drie stappen:
    1. Een "proposal method", om een verzameling van "temporal proposals", elk met een variërende duur en hun eigen start en eind tijd. Elke proposal heeft drie stages: *starting, course* en *ending*.
    2. Voor elke proposal wordt er STPP (structured temporal pyramid pooling) toegepast door (1) de proposal op te splitsen in drie delen; (2) temporal pyramidal representaties te maken voor elk deel; (3) een globale representatie maken voor de hele proposal.
    3. Twee classifiers worden gebruikt: herkennen van de actie en de "volledigheid"van de actie nagaan.

- Bron [28]
  - Temporal action detection = moet enerzijds detecteren of al dan niet een actie voorkomt, en anderzijds hoelang deze actie duurt, wat een uitdaging is bij untrimmed videos.
  - Veel moderne aanpakken gaan als volgt te werk: eerst wordt er klasse-onafhankelijke proposals gegenereerd door

- Bron [1]
  - Sliding window: laatste 30 frames bijhouden in buffer om hoge zekerheid van classificatie te voorzien; met majority voting de actie bepalen die het meeste voorkomt.
  - Classifier: random forests. Beslissingsbomen aanmaken via ID3 algoritme
- Bron [4]
  - Depth-based action recognition.
  - *key frames* worden geproduceerd uit skeletsequenties door gebruik te maken van de joints als **spatial-temporal interest points (STIPs)**. Deze worden gemapt in een dieptesequentie om een actie sequentie te representeren. De contour van de persoon wordt per frame bepaald. Op basis van deze contour en de tijd worden features opgehaald. Als classifier gebruiken ze een *extreme learning machine*
  - Voordeel van key frames: ze bevatten de meest informatieve frames. Twee methodieken om de key frames op te halen:
    1. **Interframe difference**: een nieuwe key-frame wordt gekozen als het verschil tussen twee frames een bepaald threshold overschrijft.
    2. **Clustering**: groeperen van frames die op elkaar lijken op basis van low-level features. Uit die groep wordt dan de keyframe genomen, die het dichtst bij het centrum van dat cluster ligt.
  - Zij gebruiken het 'opgenomen verschil': Een positie van een joint $P_{i,j}$ met i het frame index en j de joint index, kan gelijkgesteld worden als $P_{i,j} = x_{i,j}, y_{i,j}, z_{i,j}$

    Het opgenomen verschil is dan:

    $$D_i = \sum_{j=1}^{n} ||P_{i,j} - P_{i-1,j}||^2$$

    met $|| \cdot ||$ de euclidische afstand en n het aantal joints.
  - key frames worden dan gekozen op basis van maximum of minimum D_i binnen een sliding window. Een probleem: D_i is vrij laag voor de eerste en laatste aantal frames. De key frames worden dus eerder gecentraliseerd en kan de sequentie niet accuraat bepaalt worden. Stapsgewijze oplossing:
    1. Voor een video met N frames: neem de som van D_i van $i = 2$ tot $i = N$:

       $$D_N = \sum_{i=2}^{N} D_i$$

    2. Bepaal een aantal key frames K en bereken het gemiddelde van incrementen:

       $$D_{avg} = D_N / K$$

3. Voor $i = 2$ tot $i = L$ wordt het verschil berekent:

$$W_L = D_L - k * D_{avg}, k \in K$$

zodat er een verzameling W_L is. Het minimum van deze set wordt de key frame.

- Features op basis van contour
- Actieherkenning met neurale netwerken (EXTREME LEARNING)
- **Samenvatting:**
  - Actionherkenningsmethode voor kinect.
  - Features op basis van menselijke contour van een keyframe uit een dieptebeeld. Als constraint is er het temporaal verschil.
  - 'multi-hidden layer extreme learning machine' voor classificatie

- Bron [20]
  - Een bepaalde actie kan onderverdeeld worden in een sequentie van poses.
- Bron [29]
  - HMM laten toe om de temporale evolutie te modelleren.
  - De **feature set** en **emission probability function** moeten goed gedefinieerd zijn.
  - Gegeven een set C van actie classes $\Lambda = \lambda^1 ... \lambda^C$, zoek de klasse λ^* die de kans op $P(\lambda|O)$ maximaliseert met $O = \{o_1 ... o_T\}$ de frame observaties.
  - Een HMM voor elke actie. Classificatie voor een observatiereeks O wordt uitgevoerd door het model te pakken met de hoogste waarschijnlijkheid:

$$\lambda^* = \arg \max_{1 \leq c \leq C} \left[P(O|\lambda^c) \right]$$

  - Twee verschillende features gebruikt:
    - Projection histograms
    - Shape descriptors

5.2 Machine learning

5.2.1 Features

- Een **feature** is een individueel, meetbare eigenschap of karakteristiek van een object dat geobserveerd wordt.
- Eigenschappen:
  - Informatief: de informatiewinst van de feature moet hoog zijn

  - Discriminative: op basis van de feature moet het eenvoudig zijn om het onderscheid te maken tussen de verschillende klassen
  - Onafhankelijk: De feature op zich mag van geen andere feature of meetwaarde van dezelfde feature afhangen.
- Een **sparse feature discriptor** heeft een variabel aantal features. Een **dense feature descriptor** heeft een vast aantal features.
- **Feature extraction** (≡ dimensionality reduction) is het verzamelen van features uit ruwe data zodat deze kunnen gebruikt worden als feature vector bij een classifier.
- Een **feature vector** is een n-dimensionale vector van numerieke features.
- De **feature space** (≡ vectorruimte) beschrijft de ruimte waarin de features zich bevinden. (bv 3 verschillende features = $\mathcal{R}^3$)
- **Feature construction** is het maken van nieuwe features op basis van reeds bestaande features. De mapping is een functie ϕ, van $\mathcal{R}^n$ naar $\mathcal{R}^{n+1}$, met f de geconstrueerde feature op basis van bestaande features, bv $f = x_1/x_2$.

$$\phi(x_1, x_2, ..., x_n) = (x_1, x_2, ...x_n, f)$$

5.2.2 Classifier

- Identificeren tot welke klasse een nieuwe observatie behoort, gebaseerd op een training set waarvan de klassen wel gekend zijn.
- **Lineaire classifiers** geven aan elke klasse k een score op basis van de combinatie van de feature vector met een gewichtenvector met het scalair product. De gekozen klasse is dan die met de hoogste score. Eenvoudiger geschreven:

$$score(X_i, k) = \beta_k \cdot X_i$$

  - X_i = de feature vector voor instantie i
  - β_k = de gewichtenvector voor klasse k
- ToDo: later onderzoeken
- **Support Vector machines**
- **Random forests**
- **Boosting**

5.2.3 Hidden Markov Model

Bron [30]

Markov model

- Markov process = stochastisch process met volgende eigenschappen:
  - Het aantal toestanden $S = \{s_1, s_2, ...s_{|S|}\}$ is eindig.

– Observatie van een sequentie over de tijd $\vec{z} \in S^T$

Voorbeeld:

$S = \{sun, cloud, rain\}$ met $|S| = 3$ en $\vec{z} = \{z_1 = S_{sun}, z_2 = S_{cloud}, z_3 = S_{cloud}, z_4 = S_{rain}, z_5 = S_{cloud}\}$ met $T = 5$.

– Markov Assumpties:

1. Een toestand is enkel afhankelijk van de vorige toestand (**Markov Property**).

$$P(z_t|z_{t-1}, z_{t-2}, ..., z_1) = P(z_t|z_{t-1})$$

2. De waarschijnlijk is constant in de tijd

$$P(z_t|z_{t-1}) = P(z_2|z_1); t \in 2...T$$

– Beginstaat $z_0 \equiv s_0$.

– Transitiematrix $A \in \mathbb{R}^{(|S|+1)\times(|S|+1)}$, met $A_{ij} = P(i \to j)$, :

$$A = \begin{array}{c|cccc} & s_0 & s_{sun} & s_{cloud} & s_{rain} \\ s_0 & 0 & .33 & .33 & .33 \\ s_{sun} & 0 & .8 & .1 & .1 \\ s_{cloud} & 0 & .2 & .6 & .2 \\ s_{rain} & 0 & .1 & .2 & .7 \end{array}$$

- Twee vragen die een markov model kunnen oplossen:

1. Wat is de kans op een bepaalde toestandenvector $\vec{z}$?

$$P(\vec{z}) = \prod_{t=1}^{T} A_{z_{t-1}z_t}$$

Stel $\vec{z} = \{z_1 = S_{sun}, z_2 = S_{cloud}, z_3 = S_{rain}, z_4 = S_{rain}, z_5 = S_{cloud}\}$ dan is $P(\vec{z}) = 0.33 \times 0.1 \times 0.2 \times 0.7 \times 0.2$

2. Gegeven $\vec{z}$, hoe worden best de parameters van A benaderd zodat de kans op $\vec{z}$ maximaal is.

$$A_{ij} = \frac{\sum_{t=1}^{T}\{z_{t-1} = s_i \wedge z_t = s_j\}}{\sum_{t=1}^{T}\{z_{t-1} = s_i\}}$$

De maximale kans om van staat i naar j te gaan is het aantal transities van i naar j gedeeld door het totaal aantal keer dat we in i zitten. Met andere woorden: Hoeveel % zaten we in i als we van j komen.

Hidden Markov Model

- HMM = Veronderstelt een markov process met verborgen toestanden
- Bij een HMM: de staat is niet zichtbaar, maar de output is wel zichtbaar.
- Formeel:
  - Sequentie van geobserveerde outputs $x = \{x_1, x_2, ..., x_T\}$ uit een alfabet $V = \{v_1, v_2, ..., v_{|V|}\}$
  - Er is ook een sequentie van staten $z = \{z_1, z_2, ...z_T\}$ uit een alfabet $S = \{s_1, s_2, ...s_{|S|}\}$, maar deze zijn niet zichtbaar.
  - Transitiematrix A_{ij} wel bekend.
  - Kans dat een bepaalde output gegenereerd wordt in functie van de verborgen toestanden:

    $$P(x_t = v_k | z_t = s_j) = P(x_t = v_k | x_1, ..., x_T, z_1, ..., z_T) = B_{jk}$$

    Matrix B geeft de waarschijnlijkheid dat een verborgen toestand s_j de output v_k teruggeeft.
- Vaak voorkomende problemen die opgelost kunnen worden met HMM:
  - Gegeven de parameters en geobserveerde data, benader de optimale sequentie van verborgen toestanden.
  - Gegeven de parameters en geobserveerde data, bereken de kans op die data. → Wordt het 'decoding' probleem (**Viterbi Algoritme**) genoemd en wordt gebruikt bij continue actieherkenning.
  - Gegeven de geobserveerde data, benader de parameters van A en B.
- Het **decoding probleem**: `http://jedlik.phy.bme.hu/~gerjanos/HMM/node8.html`
  - Zoek de meest waarschijnlijke reeks van toestanden $\vec{z} \in S^T$ voor een verzameling van observaties $\vec{x} \in V^T$.
  - Hoe 'meest waarschijnlijke toestandensequentie' definiëren. Een mogelijke manier is om de meest waarschijnlijke staat s_t voor x_t te berekenen, en alle q_t dier daar aan voldoen te concateneren. Andere manier is **Viterbi algoritme** die de hele toestandensequentie met de grootste waarschijnlijkheid teruggeeft.
  - Hulpvariabele:

    $$\delta_t(i) = \max_{q_1, q_2, ... q_{t-1}} p\{q_1, q_2, ...q_{t-1}, q_t = i, o_1, o_2, ...o_{t-1} | \lambda\}$$

    die de hoogste kans beschrijft dat een partiele observatie en toestandensequentie tot $t = t$ kan hebben, wanneer de huidige staat i is.

Acroniemen

CRF Conditional Random Field. 7, 9

DTW Dynamic Temporal Warping. 7, 9

FTP Fourier Temporal Pyramid. 7

HMM Hidden Markov Model. 7, 9, 18

LDA Linear Discriminant Analysis. 7

STIP Spatio-Temporal Interest Points. 6

Bibliografie

[1] F. Deboeverie, S. Roegiers, G. Allebosch, P. Veelaert, and W. Philips, "Human gesture classification by brute-force machine learning for exergaming in physiotherapy," in IEEE Conference on Computational Intelligence and Games, CIG, 2016.

[2] M. Devi, S. Saharia, and D. D.K.Bhattacharyya, "Dance Gesture Recognition: A Survey," International Journal of Computer Applications, vol. 122, no. 5, pp. 19–26, 2015.

[3] Y. Zhu, W. Chen, and G. Guo, "Fusing spatiotemporal features and joints for 3D action recognition," IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 486–491, 2013.

[4] S. Liu and H. Wang, "Action recognition using key-frame features of depth sequence and ELM," (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 8, No. 10, 2017, 2017.

[5] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, and M. Finocchio, "Real-time human pose recognition in parts from single depth images," CVPR, 2011.

[6] Laptev and Lindeberg, "Space-time interest points," Proceedings Ninth IEEE International Conference on Computer Vision, pp. 432–439 vol.1, 2005. [Online]. Available: http://ieeexplore.ieee.org/document/1238378/

[7] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie, "Behaviour Recognition via Sparse Spatio-Temporal Features," 2005.

[8] G. Willems, T. Tuytelaars, and L. V. Gool, "An efficient dense and scale-invariant spatio-temporal interest point detector," 2008.

[9] H. Wang, A. Kl, C. Schmid, L. Cheng-lin, H. Wang, A. Kl, C. Schmid, L. C.-l. A. Recognition, and A. Kl, "Action Recognition by Dense Trajectories," Cvpr'11, 2011.

[10] W. Li, Z. Zhang, and Z. Liu, "Action recognition based on a bag of 3d points." IEEE, jun 2010, pp. 9–14. [Online]. Available: https://www.microsoft.com/en-us/research/publication/action-recognition-based-bag-3d-points/

[11] J. Wang, Z. Liu, J. Chorowski, Z. Chen, and Y. Wu, "Robust 3d action recognition with random occupancy patterns," 2012.

[12] L. Xia, C.-c. Chen, and J. Aggarwal, "View Invariant Human Action Recognition Using Histograms of 3D Joints," CVPR 2012 HAU3D Workshop, pp. 20–27, 2012.

[13] J. Gu, X. Ding, S. Wang, and Y. Wu, "Action and gait recognition from recovered 3-D human joints," IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 2010.

[14] R. Poppe, "A survey on vision-based human action recognition," Image and Vision Computing, vol. 28, no. 6, pp. 976–990, 2010.

[15] J. Wang, Z. Liu, Y. Wu, and J. Yuan, "Mining Actionlet Ensemble for Action Recognition with Depth Cameras," 2012.

[16] X. Yang and Y. L. Tian, "EigenJoints-based action recognition using Naïve-Bayes-Nearest-Neighbor," in IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2012.

[17] L. Han, X. Wu, W. Liang, G. Hou, and Y. Jia, "Discriminative human action recognition in the learned hierarchical manifold space," Image and Vision Computing, vol. 28, no. 5, pp. 836–849, 2010.

[18] M. Müller and T. Röder, "Motion Templates for Automatic Classification and Retrieval of Motion Capture Data," SCA, 2006.

[19] A. Ng and M. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," 2002.

[20] S. Carlsson and J. Sullivan, "Action recognition by shape matching to key frames," Workshop on Models versus Exemplars in Computer Vision, jan 2001.

[21] D. Weinland and E. Boyer, "Action recognition using exemplar-based embedding," 26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR, no. May, 2008.

[22] A. Fathi and G. Mori, "Human pose estimation using motion exemplars," Proceedings of the IEEE International Conference on Computer Vision, pp. 1–8, 2007.

[23] L. Wang, Y. Qiao, and X. Tang, "Action recognition and detection by combining motion and appearance features," 2014.

[24] S. M. Kang and R. Wildes, "Review of Action Recognition and Detection Methods," 2016.

[25] Y. Zhao, Y. Xiong, L. Wang, Z. Wu, X. Tang, and D. Lin, "Temporal Action Detection with Structured Segment Networks," 2017.

[26] G. Singh and F. Cuzzolin, "Untrimmed Video Classification for Activity Detection: submission to ActivityNet Challenge," arXiv preprint arXiv:1607.01979, 2016.

[27] J. Yuan, B. Ni, X. Yang, and A. Kassim, "Temporal Action Localization with Pyramid of Score Distribution Features," 2016, pp. 3093–3102.

[28] J. Huang, N. Li, T. Zhang, G. Li, T. Huang, and W. Gao, "SAP: Self-Adaptive Proposal Model for Temporal Action Detection Based on Reinforcement Learning," 2018. [Online]. Available: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16109

[29] R. Vezzani, D. Baltieri, and R. Cucchiara, "HMM Based Action Recognition with Projection Histogram Features," 2010.

[30] D. Ramage, "Hidden Markov Models Fundamentals," 2007. [Online]. Available: http://see.stanford.edu/materials/aimlcs229/cs229-hmm.pdf