FACULTY OF ENGINEERING AND ARCHITECTURE

Live actieherkenning met de Kinect sensor in Python

Bert De Saffel
Student number: 01614222

Supervisors: Prof. dr. ir. Peter Veelaert, Prof. dr. ir. Wilfried Philips
Counsellors: ing. Sanne Roegiers, ing. Dimitri Van Cauwelaert

Master's dissertation submitted in order to obtain the academic degree of
Master of Science in Information Engineering Technology

Academic year 2018-2019

GHENT UNIVERSITY

Abstract

Inhoudsopgave

Hoofdstuk 1

Methodologie

- Basisidee: *key frames* = kan zeker voordeel brengen
  - Welke acties herkennen?
  - Wat is 'te weinig verschil'?
  - Wanneer gebruikte frames weggooien?
  - Studie hidden markov model → zie 3.3.s
- Classificatiemodel pas vastleggen nadat verschillende mogelijkheden getest zijn op dataset.
  - Support vector machines
  - ensemble methoden
  - nog geen prioriteit

Hoofdstuk 2

Literatuur

- Bron [1]
  - 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:
  - 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 [2] 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 [3]

∨ Bevat bruikbare datasets van skelet-, diepte- en kleurenbeelden

- Ook hier praten ze over de vaak voorkomende uitdagingen: Intra-en interklasse variaties, de omgeving en de grootte van de verzameling van acties die er eigenlijk bestaan.
- Hier tonen ze ook weer het nut van de kinect sensor aan, en gebruiken de kinect
- Ze geven een nieuw algoritme om menselijke actieherkenning uit te voeren vanuit een dieptebeeld, een view-invariante representatie van poses en het systeem werkt real-time.

  ! Het real-time component bevat drie zaken:

  * Het verkrijgen van de 3D locaties van de joints → via bron [2]
  * Het berekenen van HOJ3D (histogram)
  * Classificatie
- Histogram gebaseerde representatie van 3D poses (HOJ3D genoemd) = partitie van 3D ruimte in n "bins", gebruik maken van een bolcoördinatensysteem. Selectie van 12 joints die een compacte representatie van het skelet weergeven. (hand en pols, voet en enkel worden gecombineerd).
- Het centrum van deze 3D ruimte is de de heup joint. Er is ook een vector α, parallel met de grond, door de heup (van links naar rechts), en een vector θ loodrecht op de grond en door het centrum Deze 3D ruimte (figuur 3b) wordt opgesplitst in n partities.

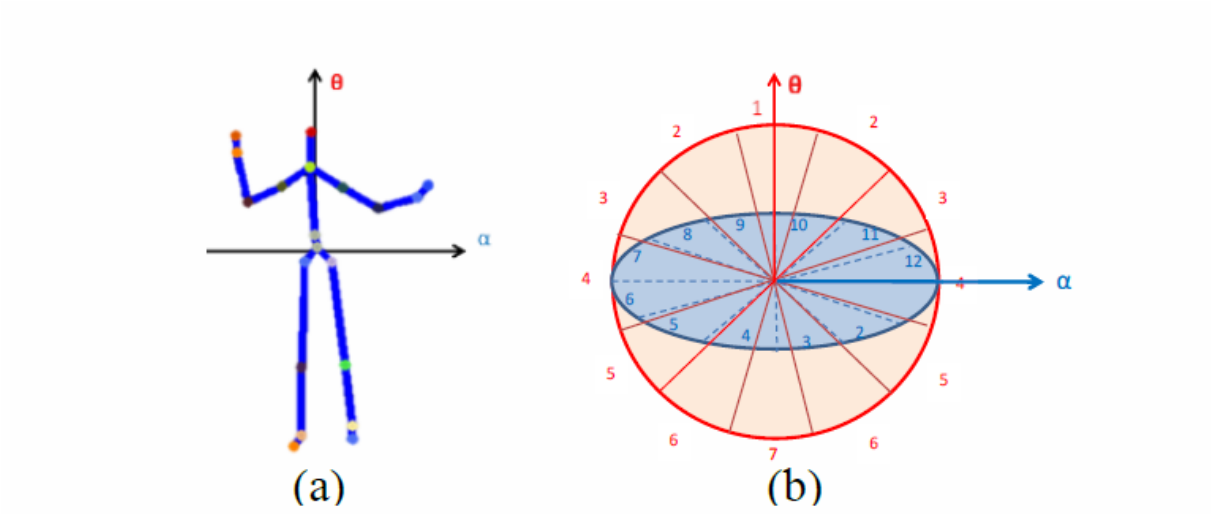

Figure 3: (a) Reference coordinates of HOJ3D. (b) Modified spherical coordinate system for joint location binning.

Voor θ: [0, 15], [15, 45], [45, 75], [75, 105] [105, 135], [165, 180]. (7 bins)
Voor α: 30 graden voor elke bin, dus 12 bins.
in totaal $7 * 12 = 84$ bins
Via deze bolcoördinaten kan elke 3D joint gelokaliseerd worden in een unieke bin

- De 3 joints die gebruikt worden om het bolcoördinatenstelsel te oriënteren staan uiteraard vast. De overige 9 joints worden onderverdeeld in één van de 84 bins.
- Om de representatie robust te maken, wordt één enkele joint over verschillende, naburige bins verdeeld (8 buren), op basis van gewichtsfunctie:

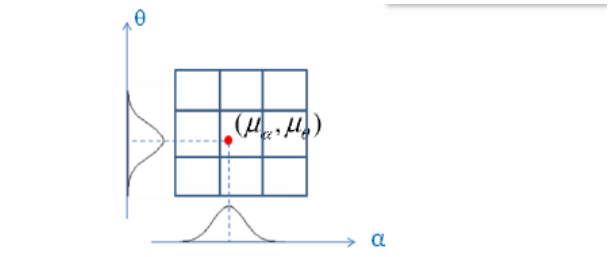

Figure 4: Voting using a Gaussian weight function.

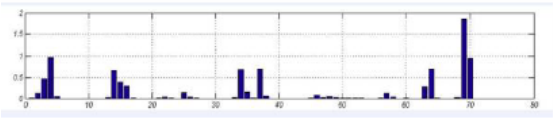
Figure 5: Example of the HOJ3D of a posture.

- Linear discriminant analysis (LDA) wordt toegepast om dominante features eruit deze histogram te halen.
- Ze stellen voor om kleurenbeelden te combineren met dieptebeelden om algoritmen te ontdekken met beter herkenning
- Ze beweren sneller te zijn dan bron [4]

- Bron [4] (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 [5]
  - 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 [6]:
      · 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 [7]:
        · 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 [8]
  - 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 [9]
  - 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 [10]
  - 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

Hoofdstuk 3

Machine learning

3.1 Features

- Een **feature** is een individueel, meetbare eigenschap of karakteristiek van een object dat geobserveerd wordt.
- Het kiezen van *informatieve, discriminative en onafhankelijke* features is belangrijk.
  - Informatief: de informatiewinst van de feature moet hoog zijn
  - Discriminative: op basis van de feature moet het eenvoudig zijn het onderscheid te maken tussen de verschillende klassen
  - Onafhankelijk: De feature op zich mag van geen andere feature afhangen.
- **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 $f = x_1/x_2$.

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

3.2 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**

3.3 Hidden Markov Model

- Markov process = stochastisc process met volgende eigenschappen:
  - Aantal toestanden is eindig.
  - Een toestand is enkel afhankelijk van de vorige toestand.
  - De waarschijnlijk is constant in de tijd.
- HMM = Veronderstelt een markov process met verborgen toestanden
- Bij een gewoon markov model: elke toestand is zichtbaar. Dus enkel de kans om van toestand x naar y te gaan zijn de enige parameters.
- Bij een HMM: de staat is niet zichtbaar, maar de output is wel zichtbaar.
- 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 genoemd en wordt gebruikt bij continue actieherkenning.
  - Gegeven de geobserveerde data, benader de parameters.
- Het **decoding probleem**: `http://jedlik.phy.bme.hu/~gerjanos/HMM/node8.html`
  - Zoek de meest waarschijnlijke toestandensequentie voor een verzameling van observaties $O = o_1, o_2, ..., o_T$ en een model $\lambda = (A, B, \pi)$.
  - Hoe 'meest waarschijnlijke toestandensequentie' definiëren. Een mogelijke manier is om de meest waarschijnlijke staat q_t voor tt 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.

Bibliografie

[1] S. Min Kang and R. Wildes, "Review of action recognition and detection methods," 10 2016.

[2] J. Shotton, A. Fitzgibbon, A. Blake, A. Kipman, M. Finocchio, B. Moore, and T. Sharp, "Real-time human pose recognition in parts from a single depth image." IEEE, June 2011, best Paper Award. [Online]. Available: https://www.microsoft.com/en-us/research/publication/real-time-human-pose-recognition-in-parts-from-a-single-depth-image/

[3] L. Xia, C. Chen, and J. Aggarwal, "View invariant human action recognition using histograms of 3d joints," in Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on. IEEE, 2012.

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

[5] Y. Zhao, Y. Xiong, L. Wang, Z. Wu, X. Tang, and D. Lin, "Temporal action detection with structured segment networks," 2017.

[6] G. Singh and F. Cuzzolin, "Untrimmed video classification for activity detection: submission to activitynet challenge," arXiv preprint arXiv:1607.01979, 2016.

[7] J. Yuan, B. Ni, X. Yang, and A. Kassim, "Temporal action localization with pyramid of score distribution features," 06 2016, pp. 3093–3102.

[8] 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

[9] F. Deboeverie, S. Roegiers, G. Allebosch, P. Veelaert, and W. Philips, "Human gesture classification by brute-force machine learning for exergaming in physiotherapy," in 2016 IEEE Conference on Computational Intelligence and Games (CIG), Sept 2016, pp. 1–7. [Online]. Available: http://ieeexplore.ieee.org/document/7860414/

[10] 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.