Kijkrichtingen detecteren door middel van elektromyografie

Kevin Boets
Katholieke Universiteit Leuven
Departement Computerwetenschappen
kevin.boets@student.kuleuven.be

Gertjan Franken
Katholieke Universiteit Leuven
Departement Computerwetenschappen
gertjan.franken@student.kuleuven.be

Abstract

In de zoektocht naar natuurlijke user interfaces gebeurt er onderzoek naar besturing door middel van de ogen. In deze paper wordt gebruik gemaakt van EMG (Elektromyografie) signalen om eenvoudige oogbewegingen te detecteren. Hiervoor onderzoeken we twee verschillende methodes: thresholds en patronen. Beide methodes worden dan ook grondig besproken en vergeleken doorheen de tekst.

1 Introductie

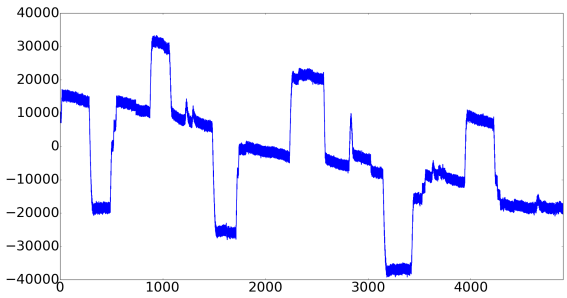
Reeds verschillende manieren om oogbewegingen te detecteren zijn bedacht. Zo kan men al door middel van camera's de ogen volgen. Hierbij gaat men op zoek naar de positie van de iris om de stand van het oog te achterhalen. Deze technieken zijn al ver ontwikkeld, maar toch gaan we nu een andere techniek onderzoeken. Aan de hand van de EMG signalen die gemeten worden door middel van sensoren die rond de ogen zijn aangebracht, proberen we te achterhalen in welke richting de gebruiker kijkt. Als invoer krijgen we twee elektrooculogrammen die respectievelijk de kijkrichtingen links-rechts en boven-onder voorstellen. Figuur 1 geeft een mooi voorbeeld weer van het elektrooculogram betreffende de kijkrichtingen links en rechts. Hierbij stellen de dalen de kijkrichting links voor en stellen de bergen de kijkrichting rechts voor.

2 Preprocessing

De signalen die we krijgen doorgestuurd, afkomstig van de sensoren, zijn zeer ruw. Om hier kijkrichtingen uit af te kunnen leiden, gaan we de data eerst beter leesbaar maken. Dit is de preprocessing-stap. Deze stap voeren we uit op elke dataset, alvorens we deze gaan analyseren. Beschouw de data in figuur 1 als de originele data.

Als eerste gaan we de ruis en knipperingen zo goed mogelijk proberen te verminderen. Dit gebeurt door middel van een low-pass filter. Deze filter verzacht alle signalen waarvan de frequentie hoger is dan onze cutoff frequentie.

Daarna focussen we ons op de biopotentiaalschommelingen. Deze schommelingen kunnen we niet op voorhand voorspellen en verschillen van persoon tot persoon. Als we deze schommelingen kunnen verminderen,

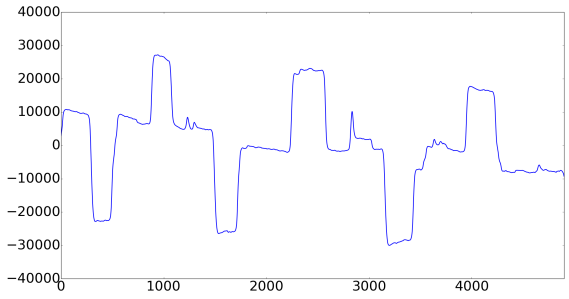


Figuur 1: Originele data rechtstreeks afkomstige van de hardware. De grootheid op de x-as is tijd en op de y-as is dit het voltage.

kunnen we meer steunen op absolute waardes van het signaal. In dit tweede deel van de preprocessing-stap gebruiken we opnieuw een low-pass filter. Deze keer gebruiken we een lage cutoff frequentie. Na deze filtering van de data, blijven enkel de signalen met een lage frequentie over. Het resultaat is de benadering van de biopotentiaal van de gebruiker die doorheen de tijd fluctueert. We kunnen nu de biopotentiaalschommelingen uit onze originele data verminderen door de gevonden benadering hiervan af te trekken. De schommelingen verdwijnen niet volledig, maar worden wel sterk verminderd. Figuur 2 stelt het resultaat van de besproken filtereringen voor.

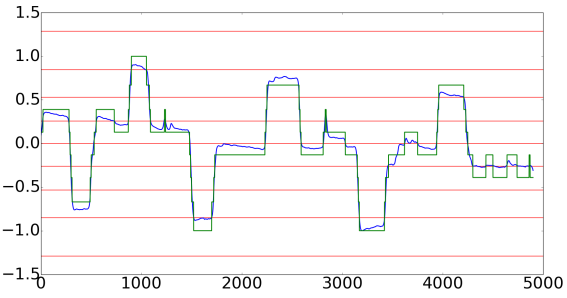
Bij veelvuldige herhaling van dezelfde kijkrichting, wordt een probleem in verband met deze methode zichtbaar. Als er veel en opeenvolgend naar éénzelfde richting gekeken wordt, zullen de waarden in de data algemeen lager of hoger zijn. De gefilterde data zal hierdoor ook algemeen lager of hoger liggen. Dit is een probleem aangezien het de benadering van de biopotentiaal moet voorstellen. Doordat de gevonden benadering foutief is, zullen ook alle volgende bewerkingen een incorrecte uitkomst hebben. Wanneer we dus deze benadering aftrekken van de oorspronkelijke data, zal de data ongewenst verschuiven volgens de y-as. Hierdoor zal het werkend met absolute waarden slechte resultaten leveren. Om dit te verkomen, zullen we er voor zorgen dat er wordt gewisseld van kijkrichting, waar mogelijk.

Na toepassing van de vorige filteringen, discretiseren we



Figuur 2: Het resultaat van de twee filteringen op de originele data. De ruis is sterk verminderd en de sterkte van de knipperingen is afgenomen. De biopotentiaalschommeling is niet helemaal weg, maar wel voldoende verminderd.

de data. Deze stap is eerder optioneel en wordt enkel in de methode met patronen gebruikt. Het grote voordeel van deze stap is dat we minder berekeningen zullen moeten doen, wat de uitvoeringstijd van het programma ten goede komt. Hiervoor gebruiken we de SAX-discretisatie (Symbolic Aggregate approXimation) [Keogh *et al.*, 2005]. De verdeling van de letters die het SAX-woord opmaken, bepalen we op de volgende manier. Om een goede verdeling te hebben, moeten de letters ongeveer even veel voorkomen. Eerst wordt de data genormaliseerd. We definiëren op voorhand hoeveel waarden een letter voor moet stellen. Dit is de constante w. De data die we willen discretiseren, verdelen we op in partities van lengte w. Elke partitie zal een bepaalde letter krijgen, afhankelijk van zijn gemiddelde. Het aantal verschillende letters dat we gebruiken, hangt af van de alfabetgrootte a. Volgens de normale verdeling wordt nu elke partitie aan een letter gekoppeld. Zie Figuur 3 voor de visuele werking. Eigenlijk stelt elke letter nu een reeks van waarden voor. Om SAX-woorden voor te kunnen stellen in grafieken, geven we elke letter ook een waarde. Dit is het gemiddelde van het interval dat de letter op zicht neemt.



Figuur 3: De groene lijn stelt de discretisatie voor van de gefilterde data die in het blauw staat. De rode horizontale lijnen geven de distributie weer van de SAX-letters.

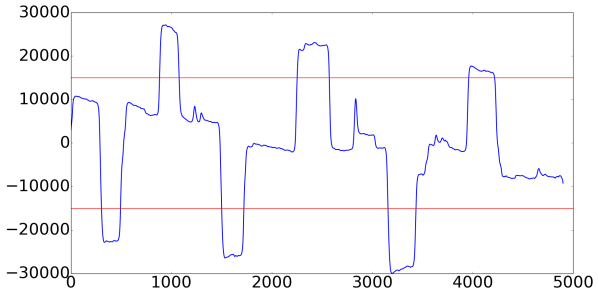
3 Gebruikte methoden

In ons onderzoek hebben we gebruik gemaakt van twee verschillende methoden om kijkrichtingen te herkennen. Deze methoden steunen respectievelijk op thresholds en patronen. Beide methoden beginnen met een calibratie-fase, gevolgd door een herkennings-fase. In de calibratie-fase laten we de gebruiker naar een aantal richtingen kijken, waarna we deze data analyseren. De gevonden informatie koppelen we aan de juiste kijkrichting. In de herkennings-fase wordt deze informatie gebruikt om in de nieuwe data kijkrichtingen te detecteren.

3.1 Thresholds

De eenvoudigste methode is de methode gebruikmakend van thresholds. Hij steunt vooral op de absolute waarden van de data. Het basis idee van deze methode is als volgt. Er wordt een bovengrens gedefinieerd voor elke kijkrichting in de calibratie-fase. Als deze overschreden wordt in de herkennings-fase, gaan we er van uit dat er in de overeenkomstige kijkrichting gekeken wordt.

In de calibratie-fase wordt aan de gebruiker gevraagd om een bepaalde sequentie van kijkrichtingen uit te voeren. Vervolgens voeren we de preprocessing in verband met de ruis, knipperingen en biopotentaail uit. Hier maken we geen gebruik van de SAX-discretisatie omdat deze method al effiiciënt genoeg is. De preprocessed dataset gebruiken we dan om goede bovengrenzen te vinden.



Figuur 4: In deze afbeelding stellen de rode horizontale lijnen de bovengrenzen voor.

Eens we de bovengrenzen hebben gevonden, kunnen we deze in de herkennings-fase gebruiken om kijkrichtingen detecteren. Om deze gevonden bovengrenzen te kunnen gebruiken, zullen we eerst dezelfde preprocessing stappen utivoeren op de nieuwe data. Belangrijk is dat we het signaal ten allen tijde stabiel houden, we werken immers met absolute waarden.

Om kijkrichtingen te detecteren, gaan we de bovengrenzen vergelijken met de waardes uit de data. Wanneer een bovengrens overschreden wordt, toont dit aan dat de gebruiker mogelijk in een van de richtingen aan het kijken is. Dit kunnen we echter niet zomaar aannemen. Een enkele piek, zoals een knippering, zou dan ook ongewild worden herkend als kijkrichting. Om voor meer zekerheid te zorgen, wachten we tot dezelfde grens opeenvolgend meerdere keren overschreden

is. Hoeveel keren definiëren we op voorhand. Het aantal dat we kiezen is een afweging tussen detectiekans en zekerheid. Bij een klein aantal is de kans op detectie hoog, maar zijn we minder zeker over de geldigheid van de detectie. Een groot aantal zal anderzijds veel zekerheid verschaffen, waarbij er wel een grotere kans is dat korte kijkrichtingen niet worden gedetecteerd. Hierbij is een goede balans belangrijk.

3.2 Patronen

In tegenstelling tot de thresholds-methode, steunt deze methode weinig op absolute waarden en eerder op relatieve waarden. Er wordt naar motieven gezocht in de calibratie-fase, die dan in de herkennings-fase met de nieuwe data vergeleken worden [Yankov *et al.*, 2007]. In deze methode maken we gebruik van SAX-discretisatie om de efficiëntie te verhogen.

Ook hier wordt in de calibratie-fase aan de gebruiker gevraagd om in een aantal kijkrichtingen te kijken. Voor elke kijkrichting hebben we nu een aantal datasets van signalen die we met elkaar gaan vergelijken. Voor elke kijkrichting zoeken we de meest succesvolle motieven. Dit zijn de motieven die het meest aantal matches hebben. We zeggen dat twee sequenties met elkaar matchen indien deze voldoende hard op elkaar lijken. Hiervoor moeten specifieke voorwaarden gedefinieerd worden, dit doen we voor ons programma in de implementatie-sectie.

In de paper [Yankov *et al.*, 2007] gebruikt men collision matrices om zo efficiënt mogelijk de motieven te vinden. Hierop hebben we ons gebaseerd en dit wordt uitgebreider uitgelegd in de implementatie-sectie.

De gevonden motieven, gekoppeld aan de bijhorende kijkrichtingen, worden doorgegeven aan de herkennings-fase. De binnenkomende data wordt verdeeld in sequenties die omgezet worden in SAX-woorden. Pas als er voldoende overeenkomst is tussen een binnengekomen SAX-woord en een SAX-woord van een motief, vergelijken we deze sequenties op preciezer niveau. Als deze sequenties weer goed genoeg op elkaar lijken, gaan we uit van de kijkrichting gekoppeld aan het motief.

4 Implementatie

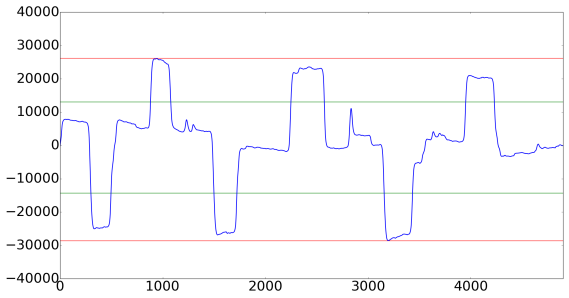
Bij het implementeren van ons programma, hebben we ons gebaseerd op de twee eerder uitgelegde methoden. In deze sectie leggen we uit hoe we de implementatie hebben verwezelijkt. Hierbij geven we ook weer hoe we bepaalde problemen hebben opgelost.

In beide methodes gebruiken we een buffer die de 1000 recentste datapunten bijhoudt. Om de 10 datapunten die we ontvangen van de sensoren, wordt deze buffer geupdate. De 10 oudste puntent worden verwijderd en de 10 recentste punten worden vooraan toegevoegd. Elke keer als dit gebeurt, voeren we ook de preprocessing stap op de data in de buffer uit. Deze buffer wordt ook getoond aan de gebruiker tijdens de uitvoering van het programma. In het begin van het programma wordt ook altijd een buffer fill uitgevoerd. Hierbij laten we de gebruiker voor zich uit kijken tot dat de buffer helemaal gevuld is.

4.1 Thresholds

In de calibratie-fase gaan we dus opzoek naar goede bovengrenzen. Nadat de preprocessing-fase is voltooid op de dataset, halen we er de kleinste en grootste waarde uit. Deze twee waarden horen elk bij een andere kijkrichting. Per kijkrichting hebben we ook een constante α waarvoor geldt: $0 < \alpha \leq 1$. We vermenigvuldigen α met het minimum of maximum (afhankelijk van de kijkrichting) en beschouwen de uitkomst als de uiteindelijke bovengrens voor de kijkrichting. Dit wordt aangetoond in figuur 5.

In ons programma hebben we α voor beide kijkrichtingen de waarde 0.5 gegeven, maar deze kan veranderd worden naar behoefte. Zo kunnen we de methode bijvoorbeeld strenger maken door α te verhogen. Het kan ook voorkomen dat het signaal bij de ene kijkrichting sterker uitwijkt dan bij de andere. In dit geval kan de α van deze kijkrichting verminderd worden.



Figuur 5: In deze afbeelding stellen de rode lijnen het minimum en maximum voor. De groene lijnen zijn het minimum en maximum vermenigvuldigd met α.

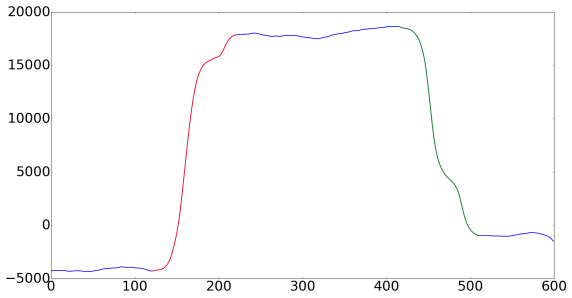
Nu we de bovengrenzen hebben gevonden, moeten we kijkrichtingen gaan zoeken in de nieuwe data. We vragen telkens de 10 recentste datapunten op van de buffer. Om de invloed van een enkele uitschieter in de data te beperken, nemen we telkens het gemiddelde van deze 10 punten. Dit gemiddelde gaan we dan vergelijken met alle bovengrenzen. Slechts wanneer de grenzen 7 opeenvolgende keren overschreden worden, zullen we beslissen dat er een kijkrichting is gevonden.

Omdat we telkens opnieuw de preprocessing-fase uitvoeren op de bufferdata, moeten we oppassen voor het probleem bij veelvuldige herhaling van dezelfde kijkrichting. Zoals eerder vermeld in de thresholds-sectie, zorgt dit voor een verschuiving van het signaal ten opzichte van de y-as. Aangezien we in de herkennings-fase weinig zeggeschap hebben over de volgorde van kijkrichtingen, zullen we dit probleem op een andere manier moeten oplossen. Hiervoor gebruiken we afvlakking van het signaal. Telkens er een threshold overschreden wordt, gaan we het gemiddelde van de vorige 990 datapunten berekenen. Vervolgens vervangen we de originele grensoverschrijdende waardes door dit gemiddelde. Door deze afvlakking, zal de benadering van de biopotentiaal eerder stabiel zijn. Het nadeel van deze techniek is dat de data destructief wordt aangepast en zo mogelijk data verloren gaat.

Dit heeft echter geen invloed voor dit systeem, aangezien we enkel geïnteresseerd zijn in de nieuwste data.

4.2 Patronen

In de calibratie-fase beschikken we over verschillende datasets die gelinkt zijn aan een bepaalde kijkrichting. Voor elke kijkrichting zoeken we minstens één paar van motieven. Deze motieven noemen we het beginmotief en het eindmotief. Een voorbeeld hiervan is te zien in figuur 6. Deze motieven nemen elk een deel van de hele kijkrichting voor hun rekening. Om telkens zo een paar te verkrijgen, leggen we een aantal voorwaarden op. Zo mogen de motieven elkaar nooit overlappen en moet elk hiervan een verschil in hoogte bestrijken. Dit laatste zorgt ervoor dat we nooit quasi horizontale lijnen, die weinig informatie bevatten, als motief zullen vinden. Wij hebben hiervoor een minimaal verschil van 5000 datapunten gekozen.



Figuur 6: In deze dataset is het beginmotief aangeduid in het rood en is het eindmotief aangeduid in het groen.

Voordat we de echte sequenties met elkaar gaan vergelijken, doen we dit eerste met hun genormaliseerde SAX-woorden. Zo kunnen we, zonder al te veel berekeningen, achterhalen welke sequenties in aanmerking komen voor een match. Hiervoor gebruiken we 10 maskers die we op voorhand hebben gedefinieerd. Zo een masker dekt telkens enkele letters van twee te vergelijken SAX-woorden af. Voor elk masker houden we ook een collision matrix bij. Die gebruiken we op de volgende manier. Indien alle overgebleven letters van de SAX-woorden overeen komen, dan wordt de overeenkomstige cel in de collision matrix geincrementeerd. Als we voor alle maskers de collision matrix hebben gemaakt, tellen we al deze matrices met elkaar op. Elk paar van SAX-woorden waarvan de waarde in de overeenkomstige cel van de resulterende matrix groter of gelijk is aan de collision threshold, komt nu in aanmerking voor een match. Deze collision threshold hebben we de waarde 7 gegeven. De oorspronkelijke sequenties van de in aanmerking gekomen SAX-woorden worden nu pas vergeleken. Dit doen we door de euclidische afstand tussen deze genormaliseerde sequenties te berekenen. Pas als deze afstand kleiner is dan de vooraf gedefinieerde range, spreken we van een match.

Tijdens de calibratie, vergroten we de collision matrix dynamisch. Dit doen we zodat na de calibratie-fase, niet alle data in één keer verwerkt moet worden, maar wel al tussen door kan gebeuren. Bij de eerste twee sequenties die worden toegevoegd, wordt nog gewoon een collision matrix opgesteld. Bij de derde en volgende sequenties, wordt er een nieuwe rij en kolom toegevoegd die overeenkomt met de nieuwe sequentie. Hiervoor worden de benodigde cellen berekend en ingevuld.

Tabel 1: Dit is een voorbeeld van een collision matrix. De letters A tot en met D stellen de verschillende SAX-woorden voor. In dit geval hebben SAX-woorden C en D de meeste collisions, namelijk 10, voor het masker van deze collision matrix.

	A	B	C	D
A	/	4	2	2
B	/	/	6	3
C	/	/	/	10
D	/	/	/	/

Nu we per sequentie het aantal matches hebben, kunnen we een rangschikking maken. We sorteren alle sequenties op het aantal matches dat ze hebben, met de sequentie met het de meeste matches vooraan. In het geval dat twee sequenties evenveel matches zouden hebben, krijgt de sequentie met de kleinste totale minimale euclidische afstand tot zijn matches voorrang.

De datapunten van de invoer worden telkens toegevoegd aan een data window. Deze window is een soort van buffer die de recentste duizend datapunten bijhoudt. Na elke nieuwe toevoeging van tien datapunten, worden de laatste honderd datapunten opgevraagd. Dit wordt gezien als een sequentie. Net zoals bij de motieven gebeurde, wordt ook voor deze sequenties gecontroleerd of ze wel een groot genoeg hoogteverschil hebben. De sequentie wordt genormaliseerd en gediscretiseerd naar een SAX-woord. Met behulp van de eerder gebruikte maskers wordt nu ook weer nagegaan of de sequentie goed genoeg lijkt op een motief. Indien de som van alle collisions per masker groter is dan de collision threshold, wordt het label van het overeenkomstige motief gegeven aan deze sequentie. Hier wordt dus enkel vergeleken op het niveau van het SAX-woord, de sequenties zelf worden niet meer met elkaar vergeleken. Pas wanneer achtereenvolgens het beginmotief en het eindmotief herkend worden, gaan we uit van een kijkrichting. Indien het beginmotief herkend wordt en de herkenning van het eindmotief op zich laat wachten, zal een timer ervoor zorgen dat er niet oneindig lang gewacht wordt op de herkenning van het eindmotief.

5 Resultaten

TODO

6 Conclusie

TODO

Acknowledgements

Onder begeleiding van Wannes Meert en prof. Luc De Raedt.
TODO

Referenties

[Keogh *et al.*, 2005] Eamonn Keogh, Jessica Lin, and Ada Fu. Hot sax: Efficiently finding the most unusual time series subsequence. In *Proceedings of the Fifth IEEE International Conference on Data Mining*, pages 226–233, November 2005.

[Yankov *et al.*, 2007] Dragomir Yankov, Eamonn Keogh, Jose Medina, Bill Chiu, and Victor Zordan. Detecting time series motifs under uniform scaling. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 844–853, August 2007.