

|  |
| --- |
|  |
| **PROJECT DATACOMMUNICATION**  **VERSLAG** |
| **DOOR KOENRAAD CHRISTIAEN** |
|  |
|  |

Contents

[Voorwoord 2](#_Toc481868978)

[Inleiding 2](#_Toc481868979)

[Technologieën 3](#_Toc481868980)

[WiiMote 3](#_Toc481868981)

[Voor -en nadelen 6](#_Toc481868982)

[Uitvoering 6](#_Toc481868983)

[UI 7](#_Toc481868984)

[Code 8](#_Toc481868985)

[Resultaten 9](#_Toc481868986)

[Conclusie 9](#_Toc481868987)

[Samenvatting 9](#_Toc481868988)

[Planning 10](#_Toc481868989)

[Bibliografie 10](#_Toc481868990)

# Voorwoord

Voor de module Datacommunication werd er gevraagd om een onbekend, nieuw stuk hardware aan te wenden om een project rond te bouwen. In dit geval werd er geopteerd voor de WiiMote controller. Deze controller werd oorspronkelijk ontworpen om via beweging games te kunnen aansturen. Het behoort toe aan de spelconsole Wii, ontworpen door Nintendo.

De communicatie van en naar de controller verloopt via bluetooth waardoor de mogelijkheid erin bestaat om de controller te verbinden met andere bluetooth ondersteunende apparaten, zoals een traditionele computer. Na een heel eind slaagde men erin alle input en output van de controller te kunnen inlezen waardoor de controller aangewend kan worden om allerhande zaken mee te besturen die niet Wii gerelateerd zijn.

Voor deze module werd er geopteerd om een bestaand spel of een derivaat van een bestaand spel te verbinden met input en output van de controller om zo gekend te geraken de manier waarop bluetooth devices met elkaar communiceren op hardware matig niveau.

In dit verslag kan de gevolgde werkwijze gevonden worden, samen met de beschrijving van de gebruikte technologieën (WiiMote, Bluetooth) en de voor -en nadelen van deze technologieën.

# Inleiding

Na enige periode van twijfel werd er gekozen om een eigen gemaakte variatie van tetris te verbinden met de WiiMote. Er werd beroep gedaan op reeds bestaande algoritmes om de eigen logica uit te bouwen. Zo kon de focus blijvend gelegd worden op de effectieve communicatie met de controller in plaats van het ontwerpen van het spel.

Eerst werd het volledige spel ontworpen met input van het traditionele keyboard. Zo kon er gegarandeerd worden dat het spel effectief op een degelijke manier werkt. Door het werken op dergelijke stapsgewijze manier kan de focus voldoende afgeschermd worden.

Na het ontwerpen van het spel dient enkel nog de input en output van de controller verbonden worden met het spel. In dit geval werd er voorzien dat het spel zowel kan bestuurd worden via het toetsenbord als de controller. Zo blijft het speel steeds speelbaar, ook zonder aanwezigheid van een effectieve controller.

# Technologieën

## WiiMote

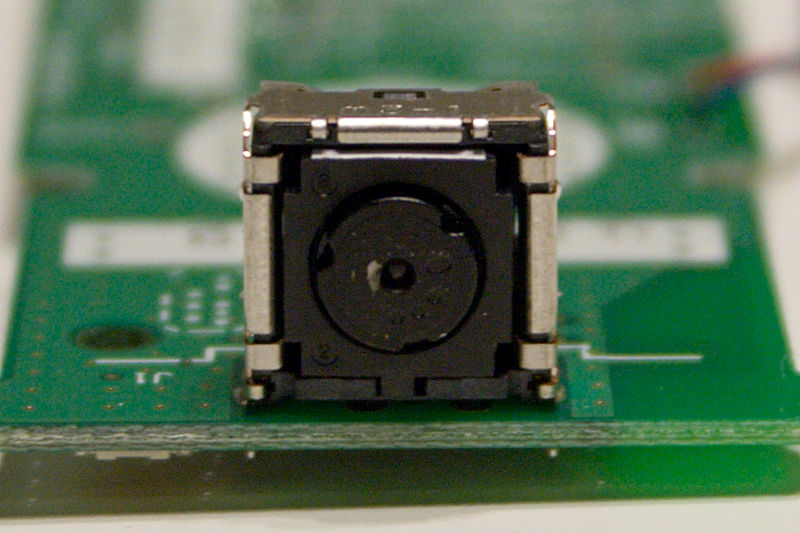
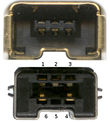
Hieronder kan de traditionele WiiMote controller gevonden worden. De input kan geleverd worden door gebruik te maken van 11 beschikbare toetsen. Daarnaast beschikt de WiiMote ook over een accelerometer die versnellingen kan meten in het 3 assenstelsel. Als laatste kan ook gebruik gemaakt worden van de infraroodcamera die aanwezig is in de top van de controller. Door deze infraroodcamera kan de WiiMote zich ook positioneren in de ruimte aan de hand van 4 warmtepunten in zijn camerabereik.

Zoals eerder vermeld kan de controller verbonden worden aan eender welk toestel aan de hand van een traditionele bluetooth verbinding.

Figuur 3: Controller layout

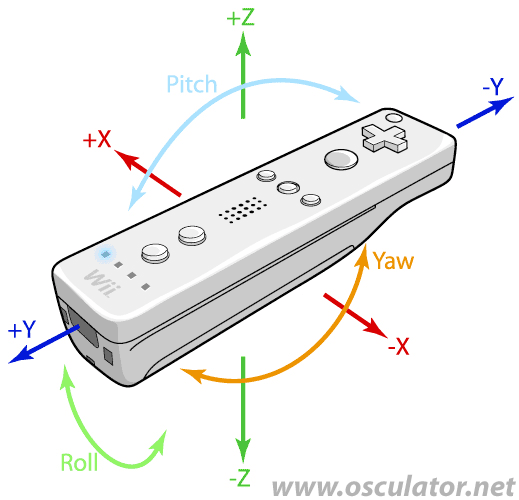
Het connecteren kan gebeuren door simultaan de knop 1 en 2 ingedrukt te houden zodat alle 4 de leds branden. Vervolgens is de WiiMote zichtbaar als bluetooth device naar andere toestellen toe. De verbinding kan tot stand gebracht worden zonder gebruik te maken van een of andere PIN-code.

Op basis van de library die verplicht te gebruiken is, kan vervolgens via de eigen code de library geïmplementeerd en aangesproken worden om data te sturen en te ontvangen.



Figuur 4: Nunchuk poort

Figuur 5: IR Camera



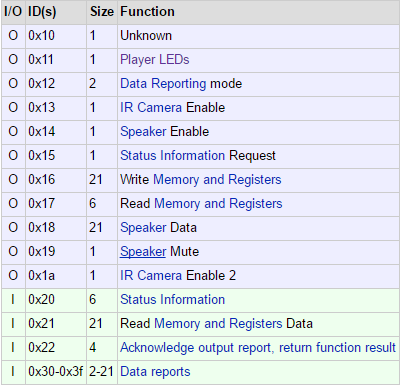
Figuur 6: Directions

Er werd geopteerd om het volledige spel enkel te besturen aan de hand van de buttons en de accelerometer. Daarbij kan zowel volledig bestuurd worden met de D-pad als met de accelerometer. Zo wordt de keuze gelaten aan de speler die steeds kan terugvallen op een van beide mogelijkheden.

De D-pad komt qua mapping volledig overeen met het intuïtieve gebruik van de pijltjestoetsen op het toetsenbord.

Via de accelerometer kan het blokje geroteerd worden door de WiiMote te roteren rond de Y-as op de bovenstaande figuur (roll). Zo kan het blokje naar rechts of naar links geroteerd worden naargelang de richting van de rotatie. Het verplaatsen van het blokje gebeurt door te roteren rond de Z-as op bovenstaande figuur (yaw).

Daarnaast worden ook de LED’s en trilfunctie van de WiiMote gebruikt om feedback te geven naar de speler toe naar aanleiding van zijn/haar spelverloop. Zo lichten de LED’s op naargelang de hoeveelheid lijnen die simultaan verwijderd worden. Bij iedere toename van de score wordt de controller ook voor een korte periode aan het trillen gebracht.

Om de input en output van de WiiMote te kunnen regelen, dient gebruik gemaakt te worden van reportdata in combinatie met een reportid.

Figuur 7: Report types

Het reportid bepaalt welk type report het is. Vervolgens kan bij elk type report een hoeveelheid data meegegeven worden in de vorm van bytes.

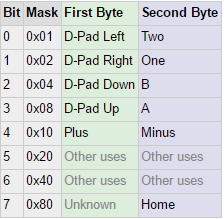
De statusreports kunnen op aanvraag opgevraagd worden om de status van de batterij, externe controllers, luidsprekers, infraroodcamera en LED’s op te vragen.

De manier waarop de datareports worden gestuurd en uit welke data deze reports bestaan, kan vooraf ingesteld worden naargelang de gegevens die de programmeur wenst te gebruiken.

In dit geval worden enkel datareports opgevraagd van de buttons en de accelerometer bij een verandering een van beide inputs. De data die vervolgens ontvangen wordt ziet er als volgt uit.

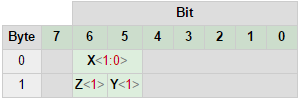


Figuur 8: Datareport structuur

Daarbij is de eerste en tweede BB respectievelijk de eerste en de tweede byte aan buttondata. Dit kan ook opgemerkt worden op de hiernaast weergegeven tabel. De status van iedere button kan opgevraagd worden door het mask toe te passen op de byte en vervolgens te kijken indien de resulterende waarde verschilt van 0.

Figuur 9: Structuur buttonbytes

Er dient echter wel opgelet te worden daar bij het ingedrukt houden de onderliggende code zal geïtereerd worden. Dit kan vermeden worden door de status van de button bij de houden en deze enkel te resetten indien de button eenmalig volledig werd losgelaten.

De X, Y en Z positie worden elk doorgegeven via een enkele byte. Daar er zowel positieve als negatieve versnellingen kunnen voorkomen wordt als nul versnelling ongeveer de waarde 0x80 genomen.

Figuur 10: X, Y, Z bytes

In dit specifieke geval bleek echter de waarde 0x79 beter te kloppen. Daar de valversnelling op aarde namelijk niet overal gelijk is, kan het voorkomen dat de nul waarde dient gekalibreerd te worden aan de hand van valversnelling die hier van toepassing is.

Door elke ontvangen waarde te verminderen met de nul waarde, bekomt men een positief of negatief getal uit naargelang de versnelling langs de specifieke as. Op basis het teken van het getal en zijn waarde, kunnen er conclusies genomen worden in welke richting de WiiMote wordt bewogen.

### Voor -en nadelen

De WiiMote is op zich een uitstekend toestel om toestellen aan te sturen op basis van beweging. Daar ondertussen de volledige structuur van de communicatie is blootgelegd, kan elk toestel aangestuurd worden door de WiiMote mits het toestel beschikt over een bluetooth connectie.

Als nadeel kan echter aangehaald worden dat de infraroodcamera zich kan focussen op verkeerde warmtepunten. Indien er geen eigen infraroodstralers voorzien worden of deze niet sterk genoeg zijn, kan de locatie anders uitvallen dan men voor ogen had. De WiiMote kan zich namelijk beginnen lokaliseren aan de hand van andere warmtebronnen in de omgeving.

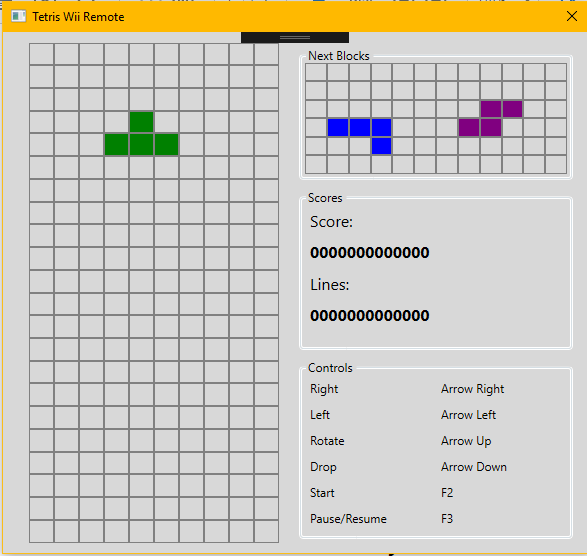
# Uitvoering

Om tetris te ontwerpen werd er zoals eerder vermeld, beroep gedaan op reeds bestaande versies van tetris. Zo werden enkele algoritmes deels overgenomen en aangepast om deze te integreren in de eigen versie. Er werd steeds een dusdanige eigen inbreng geleverd zodat er steeds ook sprake is van een eigen unieke versie.

Tetris behoort nog steeds tot de complexe categorie van de basisspellen die er op de markt zijn. De algoritmes die aangewend dienen te worden om het spel correct en vlot te laten verlopen zijn van de complexere soort. Bijgevolg dient de logica ook geoptimaliseerd te worden om met zo weinig mogelijk berekeningen het spel toch mogelijk te maken.

Als ontwikkelomgeving werd er geopteerd voor Visual Studio met als framework WPF en programmeertaal C#. De programmeertaal C# werd gekozen aan de hand van de verplicht te gebruiken library die meegeven werd om de communicatie met de WiiMote mogelijk te maken.

## UI



Figuur 1: Indeling speelveld

Het speelveld zelf werd opgebouwd via een grid in WPF. Dit grid werd opgesplitst in 10 kolommen en 22 rijen waarbij elke cel individueel kan ingekleurd worden naargelang het type blokje.

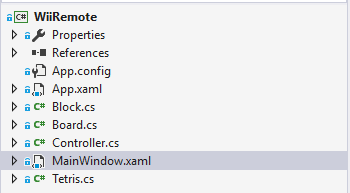
Daarnaast wordt bovenaan rechts weergegeven welke blokken er volgen nadat de huidige blok in het speelveld is geplaatst. Dit werd mogelijk gemaakt door terug gebruik te maken van een grid.

Daaronder kan de score gevonden worden in combinatie met het aantal verwijderde lijnen in dit spel. Hoe meer lijnen simultaan verwijderd worden, hoe hoger de score is die uitgedeeld wordt.

Onderaan kan vervolgens de manier van besturen gevonden worden. Deze is statisch ingesteld op de indeling van het toetsenbord daar dit spel standaard kan bestuurd worden met het toetsenbord.

Het gehele spel wordt meer dynamischer en plezieriger gemaakt door het gebruik van verschillende geluidseffecten bij verscheidene gebeurtenissen in het spel. Deze geluidseffecten werden samengesteld op basis van bestaande tetris geluiden in combinatie met geluidseffecten die niet specifiek toebehoren aan het officiële spel.

## Code

Hiernaast kunnen de gebruikte klassen gevonden worden.

Figuur 2: Code layout

Elke blok aanwezig in het spel werd opgebouwd aan de hand van de klasse “Block.cs”.

Deze klasse bevat ook alle methoden die van toepassing zijn op een blok zoals het roteren en de verplaatsing van de blok over het speelveld heen.

Vanmet een blok vast staat op het spel wordt opnieuw een nieuwe instantie gemaakt van de klasse “Block” en wordt dit bovenaan rechts visueel weergeven aan de speler als de volgende blok die beschikbaar zal worden na het spelen van de huidige blok.

In de klasse “Board.cs” kunnen alle eigenschappen en methodes gevonden worden die te maken hebben met het speelveld zelf. Deze klasse bevat een lijst met alle blokken die reeds op het speelveld en daarnaast aanwezig zijn. Hier worden ook alle controles gedaan opdat een blok in een bepaalde richting kan bewegen of roteren. Ook wordt hier de controle gedaan indien een bepaalde rij op het speelveld volledig gevuld is en verwijderd dient te worden. Deze methode wordt dan ook steeds overlopen wanneer een blok vastgehecht wordt aan het speelveld.

Daarnaast is er ook een klasse “Tetris.cs” aanwezig die de spontane voortgang van het spel garandeert op basis van een bepaald tijdsinterval en een instantie bevat van de klasse “Board.cs”. Ook wordt het bijhouden en weergeven van de scores overgelaten aan deze klasse. De methodes die instaan om het spel te starten, te stoppen en/of te pauzeren kunnen hier ook gevonden worden.

Als laatste hebben we een klasse “Controller.cs” die alle methodes bevat die het besturen met de WiiMote moeten mogelijk maken aan de hand van het implementeren van de gegeven library. Deze klasse bevat een instantie van “Tetris.cs” om gemakkelijk het reeds bestaande spel te verbinden aan de nieuwe input en output. Een volledige bespreking van communicatie met de WiiMote kan in het volgende onderdeel gevonden worden.

# Resultaten

Het spel zelf is volledig af in zijn basis. Alle basis functionaliteiten die gevonden kunnen worden in het originele spel, zijn ook hier aanwezig in het eigen ontworpen spel.

Echter ontbreken er nog enkele functionaliteiten die het spel een stuk zouden verbeteren. Zo wordt op dit moment geen score bijgehouden waardoor alle behaalde doelen niet worden opgeslagen bij het afsluiten van het spel.

Ook is er op dit moment geen mogelijkheid voorzien om de toegevoegde speelmuziek en geluidseffecten uit te schakelen op eigen wens. Dit kan bij de huidige versie enkel gedaan worden door het systeemgeluid volledig uit te schakelen.

Op vlak van besturing werd alle input van de controller optimaal benut waardoor er geen noodzaak was om de IR camera aan te wenden. Alle besturing kon gebaseerd worden op zowel de buttons als de accelerometer.

De optimale parameters bij de controle algoritmes om de juiste beweging te detecteren van de accelerometer zijn echter moeilijk correct in te stellen waardoor er vaak verkeerde of zelfs geen bewegingen worden gedetecteerd. Dit leidt vaak tot het verkeerd positioneren van een blokje in het speelveld wat vaak het einde betekent in een traditioneel tetrisspel.

# Conclusie

Het tetrisspel blijkt vrij moeilijk uit te vallen indien het bestuurd wordt aan de hand van de accelerometer van de WiiMote. Daar het al dan niet functioneren van de WiiMote als controller bij tetris sterk afhangt van de mate waarin de speler de gevraagde beweging exact uitvoert, kunnen er al gauw frustraties ontstaan bij het spelen ervan.

Daarnaast werd het door een ongelukkig geplande update van Windows 10 voor enige tijd onmogelijk gemaakt om de WiiMote te verbinden met een Windows toestel. Hierdoor kon na het uiteindelijk oplossen van dit probleem nog weinig tijd gespendeerd worden aan het optimaliseren van de accelerometer input.

Dit alles leidt tot een onaangenaam spelverloop indien de accelerometer van de WiiMote aangewend wordt om het spel te sturen. Alle andere vormen van input van zowel de controller als het keyboard blijken het spel echter wel op een correcte manier te sturen.

Mits enkele kleine toevoegingen zoals het bijhouden van scores en een zekere controle over het geluid, kan dit spel zeker als zelfstandig tetrisspel gespeeld worden op elk ondersteund toestel.

# Samenvatting

Het vervolledigen van dit project heeft heel wat inzichten geboden inzake de werking van hardware. Ook al werd er gebruik gemaakt van een library die de communicatie voor een groot stuk op zich neemt, alsnog leert de student zeker en vast werken met externe hardware.

Het verruimt de blik van mogelijkheden om producten te ontwikkelen. Zo is bewezen dat niet elke ontwikkelde software bestuurd dient te worden via de klassieke inputkanalen zoals toetsenbord en muis.

De beperking van het project om de input aan een spel te verbinden beperkt echter terug deze blik. Bijgevolg had ik graag naar de toekomst toe gezien dat de input aan gelijk welke software kan verbonden worden voor dit project. Zo kan de student meer zijn eigen inbreng leveren en zal hij/zij hoogstwaarschijnlijk een betere intrinsieke motivatie vertonen.

# Planning

|  |  |
| --- | --- |
| **Tijd (uren)** | **Uitgevoerd onderdeel** |
| 6u | Ontwerpen Tetris |
| 4u | Verbinden WiiMote |
| 4u | Opstellen verslag |

# Bibliografie

110px-Conectores\_nunchuk\_wiimote.png (110×122). (n.d.). Retrieved May 6, 2017, from <https://upload.wikimedia.org/wikipedia/commons/thumb/a/a2/Conectores_nunchuk_wiimote.png/110px-Conectores_nunchuk_wiimote.png>

camera.jpg (800×533). (n.d.). Retrieved May 6, 2017, from <http://ahmetozkurt.net/mts307/2012sp/ugur/camera.jpg>

faq:pry-wiimote.gif (519×500). (n.d.). Retrieved May 6, 2017, from <https://osculator.net/doc/_media/faq:pry-wiimote.gif>

Wiimote - WiiBrew. (n.d.). Retrieved May 6, 2017, from http://wiibrew.org/wiki/Wiimote

wiimote-762302.jpg (390×332). (n.d.). Retrieved May 6, 2017, from <https://www.wired.com/images_blogs/gadgetlab/2010/10/wiimote-762302.jpg>