Examensarbete inom Medicinsk Teknik

Grundnivå, 15 HP.

AI-assisterad spårning av flygande objekt och distansberäkning inom kastgrenar

JESPER ERIKSSON

FREDRIK JONSSON

STOCKHOLM, SVERIGE 2022

AI-assisterad spårning av flygande objekt och distansberäkning inom kastgrenar

AI-assisted Tracking of Flying Objects and Distance Measuring within Throwing Sports

Jesper Eriksson

Fredrik Jonsson

Examensarbete inom Medicinsk teknik
Grundnivå, 15 HP
Handledare på KTH: Jonas Willén
Examinator: Mats Nilsson

TRITA-CBH-GRU-2022:079

KTH
Skolan för kemi, bioteknologi och hälsa
141 52 Huddinge, Sverige

Sammanfattning

Detta examensarbete har utförts under tio veckor på uppdrag av företaget BitSim NOW. Den manuella metod som idag används för mätning av stötar inom kulstötning kan utgöra en risk för felaktiga resultat och personskador. Med hjälp av tekniska hjälpmedel kan en lösning med noggrannare mätningar och lägre risk för skador implementeras i sporten kulstötning.

Denna rapport presenterar en lösning som med hjälp av artificiell intelligens identifierar kulan utifrån en filmsekvens. Därefter beräknas längden av stöten med hjälp av en formel för kastparabeln. Lösningen jämförs sedan med en metod utan artificiell intelligens för att fastställa den bästa av de två metoderna. De variabler som jämfördes var noggrannheten på stötens längd och hur bra de två olika metoderna spårade kulan. Resultatet analyserades i relation till de uppsatta målen och sattes därefter in i ett större sammanhang.

Nyckelord
Objektdetektering, objektspårning, datorseende, kulstötning, artificiell intelligens

Abstract

This thesis project has been done during ten weeks on behalf of the company BitSim NOW. The current method used to measure the length of shot-puts presents a risk of inaccurate results along with the risk of injury for the measuring personnel. With the help of technical aids, a solution with more accurate measurements and a lower risk for injuries could be implemented in the sport of shot-puts.

This report presents a solution using artificial intelligence to first identify the shot in video films and secondly calculate the length using mathematical formulas. The solution is then compared to a method that does not use artificial intelligence, to determine what method is the superior one. The parameters that were compared were the accuracy of the length and the quality of the tracking. The result was analyzed in relation to the aims of the project and then put into a larger context.

Keywords
Object detection, object tracking, computer-vision, shot-put, artificial intelligence

Förkortningar och tekniska begrepp

AI – Artificiell intelligens. Algoritmer som försöker efterlikna mänskligt uppfattande och besluttagande.

ML – Maskininlärning. Processen som används för att lära upp AI.

CNN – Convolution Neural Network. En typ av maskininlärning som bygger på lager av noder.

FCL - Fully-connected layer. Det sista lagret i ett Convolution Neural Network.

OpenCV – Programvarubibliotek till programmeringsspråk.

CSRT – Channel and Spatial Reliability Tracker. Spårningsmetod som finns tillgänglig i OpenCVs bibliotek.

KCF – Kernelized Correlation Filter. Spårningsmetod som finns tillgänglig i OpenCVs bibliotek.

TLD – Tracking, Learning and Detection. Spårningsmetod som finns tillgänglig i OpenCVs bibliotek.

Python – Objektorienterat programmeringsspråk.

CSV – Comma Separated Values. Filformat där värdena separeras med kommatecken.

Innehållsförteckning

1 Inledning

Detta är ett examensarbete på 15 högskolepoäng utfört av två studenter från Högskoleingenjörsprogrammet i Medicinsk Teknik på Kungliga Tekniska Högskolan.

1.1 Bakgrund

Idén bakom arbetet kommer från Niclas Jansson som är anställd på företaget BitSim NOW. Projektarbetet utfördes på uppdrag av och i samarbete med företaget. De är verksamma inom elektronik och utvecklar avancerade inbäddade system och andra elektroniska system åt kunder. I en tid av många lättillgängliga tekniska hjälpmedel ser BitSim NOW en potential i att med hjälp av två studenter utveckla en produkt för distansmätning inom sporten kulstötning.

1.2 Problemformulering

Kulstötning är en kastgren inom friidrott där målet är att stöta en kula så långt som möjligt. Kulan som stöts väger 4 kg för damer och 7.26 kg för herrar och de avgörande faktorerna för stötens längd är hastigheten och vinkeln vid stöten. Underlaget som kulan landar på kan variera beroende på om det sker inomhus eller utomhus. Inomhus används oftast grus eller en form av plastmatta som underlag, utomhus används oftast grus alternativt gräs.

Längden på stöten mäts idag av en funktionär som markerar där kulan landar men i en tid av många lättillgängliga tekniska hjälpmedel finns det en potential i att digitalisera mätningen för att uppnå ett mer tillförlitligt resultat. Även inom andra kastgrenar som till exempel diskus, slägga eller spjut sker mätningen idag manuellt. Särskilt i spjut där redskapen kan flygga uppemot 100 meter kan mätningen vara särskilt besvärlig [1].

Mätningen går till på så vis att en funktionär noterar var kulan slår i marken och, från utslagsplatsen till nedslagsplatsen, mäter längden för stöten med hjälp av ett måttband eller en laser. I detta förfarande har två problem identifierats som ligger till grund för detta projekt.

Det första problemet utgörs av den mänskliga faktorn. Eftersom mätningen idag görs manuellt är den beroende av att funktionären som mäter hela tiden är fokuserad. En tävling kan pågå i flera timmar och vid en stöt befinner sig kulan i luften i mindre än 4 sekunder innan nedslaget. Under en lång tävling finns risken att den som mäter är ouppmärksam i några sekunder för att då missa var kulan har landat. Mätfel som uppstår och inte upptäcks i friidrottssammanhang riskerar att ge orättvisa resultat och kan sabotera för professionella utövare vars leverne beror av goda resultat [2].

Eftersom mätningen görs manuellt behöver funktionärer som vill minimera risken för mätfel befinna sig i närheten av kulans nedslagsplats. Som en följd av detta

uppstår det andra problemet som identifierats vilket är att den som mäter riskerar att träffas av en massiv metallkula. Att träffas av en tung projektil kan leda till allvarliga kroppsskador och kan i värsta fall vara livshotande [3].

1.3 Målsättning

Projektets mål är att realisera och utveckla ett system som på avstånd och i realtid kan mäta längden av en stöt i kastgrenen kulstötning. Syftet med arbetet är att öka tillförlitligheten till mätresultatet samt reducera risken för skador hos funktionärer.

De specificerade kraven på det slutgiltiga systemet är:

- Automatisk identifiering av stötkulan
- Följa kulans bana i luften.

Delmål för systemet är:

- Beräkna och presentera ett mätresultat inom tio sekunder.
- Presentera ett mätresultat med en felmarginal på max fem centimeter (cm).

1.4 Avgränsningar

Detta examensarbete är tidsmässigt begränsat till tio veckors heltidsstudier för två studenter. På grund av den begränsade tiden är också projektet avgränsat till att endast omfatta kastgrenen kulstötning. Projektet har också en begränsad budget vilket gör att avancerad utrustning inte kan användas.

1.5 Författarnas bidrag till examensarbete

I en del av det lösningsförslag som BitSim NOW tagit fram ingick användandet av en artificiell intelligens. Kunskapen som krävdes för att programmera den bakomliggande koden till den har Julius Kviman bidragit med.

2 Teori och bakgrund

Detta kapitel ger en bakgrund till den teoribildning som ligger till grund för rapporten. Inledningsvis beskrivs de grundläggande delarna för projektet som artificiell intelligens, maskininlärning, datorseende och objektdetektering. Därefter ges en beskrivning av tidigare arbeten och metoder för objektspårning inom andra sporter.

2.1 Maskininlärning och Artificiell Intelligens

AI eller artificiell intelligens är en term som används för att beskriva en uppsättning datoralgoritmer som efterliknar mänsklig intelligens och mänskligt beslutsfattande. Målet med en artificiell intelligens är att den ska kunna fatta beslut och lösa problem utifrån tidigare erfarenhet. Detta görs genom maskininlärning, vilket kan sägas vara den process av vilken den artificiella intelligensen tillägnar sig erfarenhet och kunskap. En typisk maskininlärningsalgoritm består av tre steg och använder statistik och metoder som analyserar datauppsättningar för att identifiera mönster [4][5]. Nedan följer en förenklad förklaring av processen:

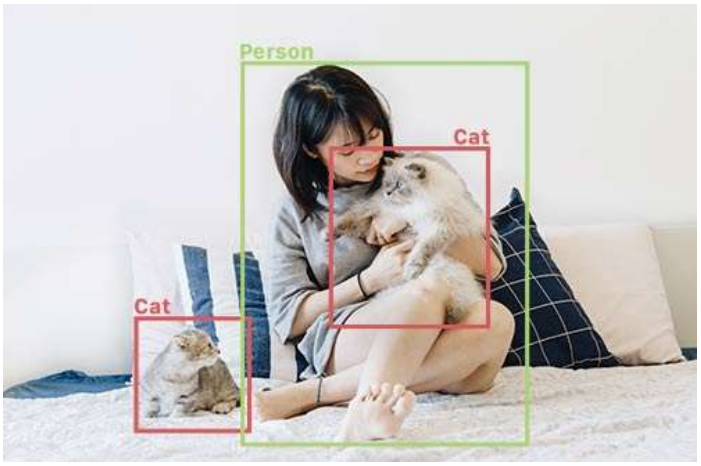
1. Beslutsprocess – Ett antal beräkningar av insignalsdata som returnerar ett antagande beroende av vilken typ av mönster som algoritmen är programmerad för att identifiera. Exempel på mönster kan vara olika färger, kanter eller andra geometriska former.
2. Felfunktion - En metod som jämför hur nära antagandet var jämfört med det kända svaret.
3. En förbättringsprocess – Används för att reducera skillnaden mellan det antagna svaret och det kända svaret genom att uppdatera hela processen [4][5].

2.2 Datorseende och Objektdetektering

Datorseende är ett fält inom datavetenskap som med hjälp av artificiell intelligens gör det möjligt för datorer att utvinna relevant och meningsfull information utifrån bilder, filmer och andra typer av visuella insignaler. Målet är att efterlikna hur den mänskliga synen fungerar. Människan kan bland annat snabbt skilja mellan olika objekt, uppskatta distans, uppfatta rörelse och hastighet [6].

Tekniken har många tillämpningsområden och återfinns bland annat inom tillverkningsindustrin där den används för att automatiskt detektera fel och brister i produktionen. Datorseende kan även appliceras på filmer och bilder för att upptäcka, spåra eller detektera objekt, detta kallas objektdetektering. Detta uppnås genom att låta algoritmen för datorseende analysera tusentals bilder och lära sig känna igen vissa mönster som är specifika och unika för ett visst objekt [7]. De algoritmer som används heter CNN och beskrivs i avsnitt 2.3. Efter inlärningsfasen kan algoritmen

appliceras och användas för nya bilder eller filmer och genom den tidigare erfarenheten känna igen objekt [7].



Figur 2.1: Objektdetektering [8].

I figur 2.1 ses en person och två katter som har identifierats med hjälp av artificiell intelligens.

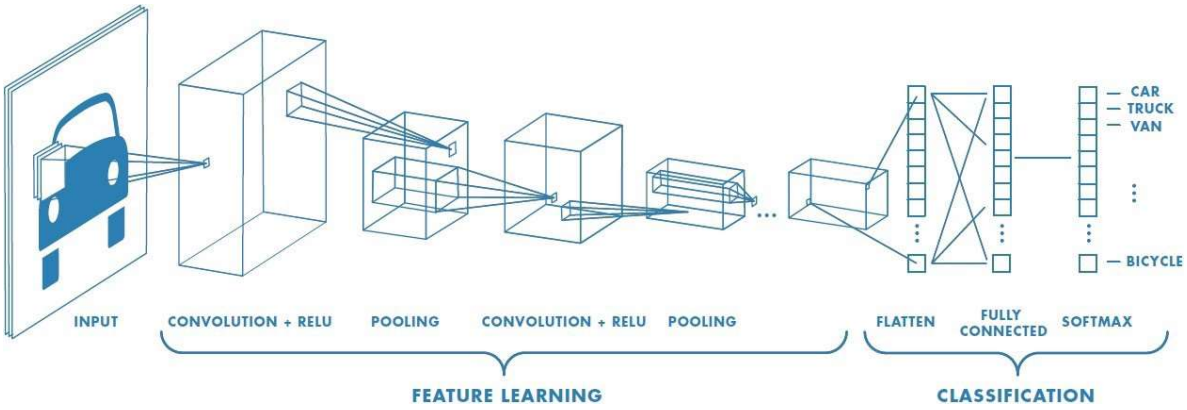
2.3 CNN

Convolutional neural network eller CNN är en typ av maskininlärning som är vanlig för analys av bilder och ljud. Det neurala nätverket är uppbyggt av olika lager där varje lager består av sammanlänkade algoritmer som kallas noder. Varje nod i ett lager har ett förutbestämt tröskelvärde och är sammankopplad med en nod i det tidigare lagret. Om utsignalen för en nod blir större än nodens tröskelvärde aktiveras den och skickar data till noden i nästkommande lager i nätverket [9].

CNN utvinner egenskaper från bilderna genom att använda filter med faltning över bilden. Faltning innebär att filtren används på en grupp av pixlar i bilden. Idén går ut på att varje filter försöker identifiera ett specifikt mönster i bilden. Resultatet av detta blir en utsignalsbild som representerar mönstret som filtret var designat att identifiera. Varje lager består av nya filter som appliceras på utsignalsbilderna från det tidigare lagret. Detta tillvägagångssätt extraherar komplexa kombinationer av mönster från originalbilden [10].

Det sista lagret före utsignalslagret kallas FCL. Detta lager gör det möjligt att förena olika kombinationer av komplexa mönster extraherade från bilden till ett nytt lager av egenskaper [10].

Nätverket består av tre grundläggande lager. Se flödesschema för CNN i figur 2.2



Figur 2.2: Flödesschema för CNN [12].

2.3.1 Convolutional-lager

Lagret kan ses som själva grundstommen i nätverket. Ett convolutional lager består av ett antal filter som appliceras på alla insignalsbilder och producerar en utsignalsbild per filter och insignalsbild. Antalet utsignalsbilder är inte beroende av mängden av kanaler i insignalsbilden. En RGB-bild har tre kanaler. Varje filter består av separata versioner för varje kanal och utsignalsbilderna kommer summeras för att producera en kanal med en utsignalsbild per filter [10].

Om det skapas en separat utsignalsbild från varje kanal skulle antalet bilder växa exponentiellt ju djupare in i nätverket som bilden skickas. Till exempel skulle en insignals RGB-signal med tio filter i första lagret producera 30 utsignalsbilder. Om nästa lager består av 30 filter skulle utsignalen bli 900 bilder. Om detta mönster upprepas skulle antalet bilder som produceras bli för krävande för att träna modellen. Varje nod i lagret utgörs av en egenskapsmatris och ett filter som tar emot en insignal. Med hjälp av filtret jämförs därefter insignalen med nodens egenskapsmatris och detta resulterar i en aktivering. Därefter används samma filter för olika insignaler som då skapar flera olika aktiveringar. Dessa aktiveringar kan sedan samlas i en utsignalsmatris där resultatet kan jämföras med det ursprungliga resultatet. I resultatet lagras bildens egenskaper, till exempel kanter och former [10].

2.3.2 Pooling-lager

Mellan varje convolutional-lager finns ett pooling-lager. Lagrets uppgift är att komprimera storleken av utsignalsbilderna från det tidigare lagret. Detta görs genom att i utsignalsbilden ersätta grupper bestående av flera pixlar till endast en pixel och på så vis reducera antalet pixlar i en bild. Tack vare detta minskas minnesanvändningen och datorkraften som används vid upplärningen av en AI. Detta kan göras med olika typer av metoder men vanligtvis används maxvärde, medelvärde eller median på pixelvärdet i det valda området. Storleken på området för varje metod kommer bestämma minskningen i pixelstorlek på bilden. Till exempel om ett område på 2x2 pixlar skulle ersättas med en pixel skulle upplösningen minska med

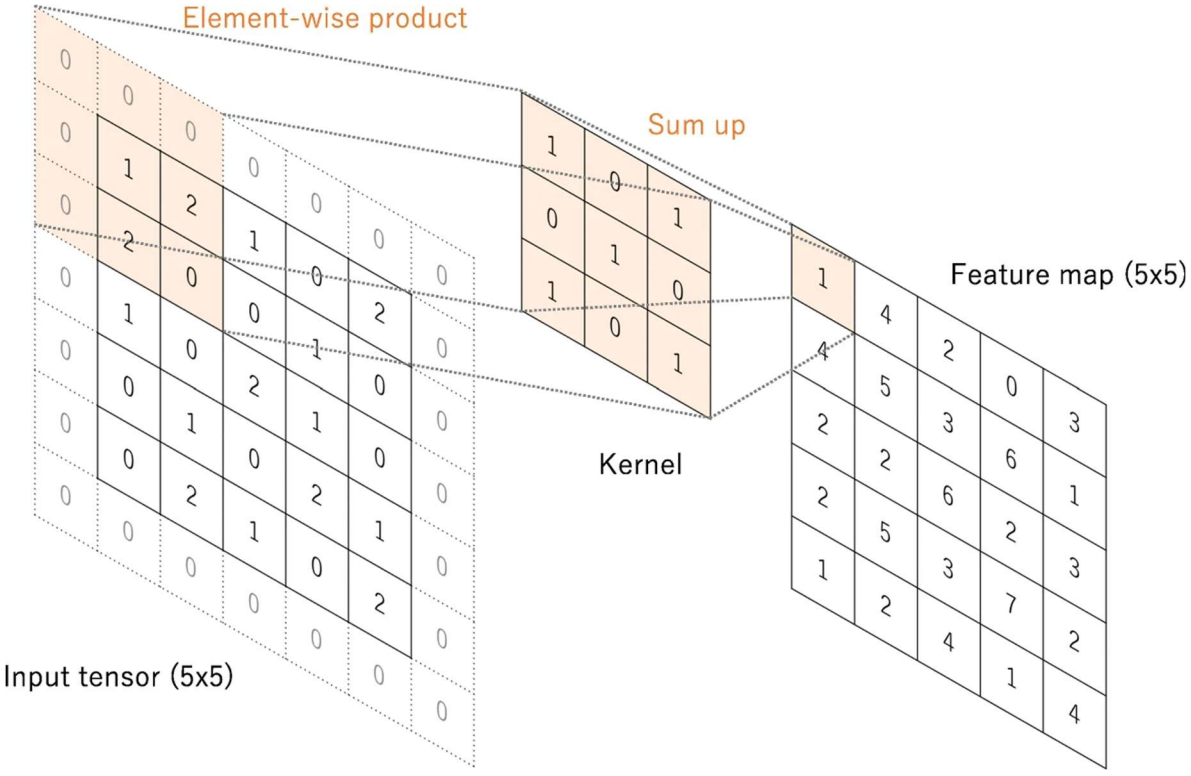
hälften i varje dimension. Detta skulle sänka storleken av bildens data med en faktor av fyra. Lagret används för att filtrera bort oönskade pixlar vilket minskar komplexiteten hos bilden och ökar effektiviteten av processen [9].

2.3.3 FCL

FCL-lagret utför klassifikation av data baserat på egenskaper som har extraherats från de tidigare lagren och de tidigare filtren [9].

2.3.4 Padding

Ett problem med att applicera faltning på en bild är att beroende av storleken på filtret blir några pixlar på kanten av bilden försummade. Detta är på grund av att mittpunkten av faltningsmatrisen inte kan bli placerad utanför bilden där det inte finns några pixlar att applicera faltning vid. När storleken på matrisen ökas kommer fler pixlar att försummas. En lösning på detta är att applicera en metod som kallas padding, vilket är en teknik som lägger till extra pixlar på kanten av bilden. Den grundläggande idén är att addera virtuella pixlar utanför originalbilden för att möjliggöra faltning på alla pixlar. För att minimera de destruktiva effekterna av att addera egenskapade pixlar kan olika metoder användas vid bestämmandet av de virtuella värdena. Ett tillvägagångssätt är att addera nollor som visas i figur 2.3, ett annat är att spegla pixlarna från insignalsbilden eller att använda pixelvärdet vid kanten [13] [14].



Figur 2.3: Zero Padding filter i ett CNN [15]. Godkänd för användning av källan.

2.4 Tidigare arbeten

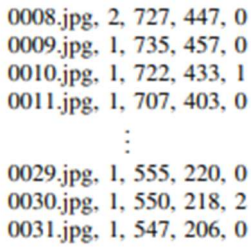
Avsnitt 2.4.1 och 2.4.2 innefattar färdiga metoder som används i nuläget för andra sporter där ett klotformat objekt spåras.

2.4.1 TrackNet

TrackNet är en teknik som utvecklades för att kunna spåra tennisbollar i racketsporten tennis. Tekniken är en typ av CNN som arbetar med datorseende. Med hjälp av en värmekarta baserad på djupinlärning kan systemet identifiera tennisbollen utifrån en ström av bilder. Dessutom har TrackNet förmågan att lära sig tennisbollens projektilbana vilket gör att programmet kan lokalisera bollen även när den är skymd av en spelare eller andra objekt [16].

Den data som ligger till grund för TrackNet samlades in genom att analysera herrarnas singel final för turneringen 2017 Summer Universiade. Filmkvalitén var 1280 x 720, bildfrekvens var 30 bilder per sekund och längden var 75 minuter. Från turneringen användes 81 matchrelaterade klipp och varje klipp innehåller ett helt spel, från serve till poäng. Totalt användes 20 844 bilder från videoklippen. Varje bild analyserades och därefter sparades nedanstående fyra egenskaper ned [16].

1. Namnet på bildrutan - Bilden sparades i formatet .jpg.
2. Synligheten av bollen - Synligheten sparades som noll om bollen inte kunde detekteras i bilden. Om bollen kunde detekteras i bilden sparades det som en etta och om bollen detekterades men ej syntes sparades det som en tvåa. Vid tvåa går det att anta var bollen är genom att använda sig av de angränsande bilderna
3. Pixelkoordinaten för bollen - Pixelkoordinaten ska optimalt vara i bollens centrum.
4. Projektilbanans mönster – Hur bollen rörde sig delades in i tre kategorier. Flygande, som sparades som en nolla. I kontakt med racket, som sparades som en etta, och studsande, som sparades som en tvåa. Figur 2.4 visar hur data sparades vid ett tennispoäng [16] [17].



Figur 2.4: Dataset från TrackNet [16]. Godkänd för användning av källan.

2.4.2 Toptracer

Toptracer är en teknik som använder datorseende för att spåra golfbollar. Systemet fungerar med hjälp av samverkande kameror och sensorer som skapar en digital tredimensionell karta över utslagsplatsen. Tack vare de ljuskänsliga sensorerna noteras golfbollens reflekterande ljusvågor i den digitala kartan och på så sätt spåras bollen. Tekniken har utöver att spåra bollen också möjlighet att utvinna annan relevant data så som den totala sträckan, bollens hastighet och utgångsvinkeln för slaget. Spårningen sker i realtid vilket gör att golfaren kan följa golfbollen och dessa data via en skärm direkt efter slaget. Toptracer lanserades och gjorde TV-debut år 2008 [18].

I figur 2.5 visas ett exempel på hur Toptracer används vid professionella turneringar.

Figur 2.5: Toptracer i användning vid PGA Championship 2020 [19].

2.5 Python

Python är ett interpreterande programmeringsspråk som lanserades år 1991 av Guido van Rossum och idag underhålls det av Python Software Foundation (PSF) [20]. Ett interpreterande språk innebär att datorn tolkar och översätter koden till maskinkod samtidigt som koden körs. Detta skiljer sig från ett kompilerat programmeringsspråk där koden på förhand översätts till maskinkod i en separat process [21].

Python utvecklades med öppen källkod vilket betyder att alla användare har tillgång till källkoden och detta ger användare möjligheten att själva vidareutveckla språket [22]. Tack vare den öppna källkoden finns flera tredjepartsbibliotek att tillgå med tillämpningar inom bland annat datorseende, objektdetektering och maskininlärning.

Python har med sin goda läsbarhet, enkla syntax och det stora standardbiblioteket tillsammans med många tredjepartsbibliotek utvecklats till att bli ett av de mest populära programmeringsspråken [23].

2.6 OpenCV

OpenCV (Open Source Computer Vision) är ett programbibliotek med öppen källkod som innehåller metoder för att bygga och utveckla en AI. OpenCV utvecklades av Intel och är ett multiplattformsbibliotek som är gratis både för kommersiellt och akademiskt syfte. Biblioteket kan appliceras i C++, C, Python, Java och MATLAB.

Biblioteket innehåller funktioner för datorseende, maskininlärning och bildbehandling. Med biblioteket är det möjligt att behandla bilder och videor för att identifiera objekt, ansikten, handskrift mm. Det finns väldigt många applikationer i OpenCV som eventuellt är relevanta för projektet, Till exempel rörelseförståelse och rörelsespårning. Tre objektidentifieringsmetoder som använder datorseende i OpenCV är KCF, TLD och CSRT [24][25].

2.6.1 KCF

KCF fungerar genom att träna ett filter med perioder som omfattar objektet och sedan träna filtret med perioder som inte omfattar objektet. Denna metod är den snabbaste metoden jämfört med TLD och CSRT, KCF anpassar sig även till skala och rotation. Eftersom KCF är tränad på en sekvens av bilder är den inte anpassad till stora rörelseskillnader [25].

2.6.2 TLD

TLD fungerar genom att träna en algoritm som sätter data i klasser och sedan kan användas för att åter detektera objektet och korrigera spårningsfel. Metoden är användbar när objektet som ska spåras är skymt vid delar av filmningen [25].

2.6.3 CSRT

CSRT-spåraren verkar genom att träna ett filter med komprimerade egenskaper. Filtret söker sedan i området runt den sista kända positionen för objektet bild för bild. CSRT är mer precis än TLD och KCF och är bra på att hantera högre hastigheter på objektet. CSRT är även en metod med ändringsbara parametrar för att optimera spårningen och kan rapportera direkt om spårningen av objektet misslyckas [25].

2.7 Matematiska formler för kastparabel

Genom att använda olika matematiska formler kan sträckan för ett kast vid en bestämd höjd med ett klotformat objekt beräknas. Med antagandet att initiala hastigheten, initiala vinkeln vid kastet och att initiala höjden är kända kan följande härledas.

Horisontala hastigheten:

$$Vx = V \cdot cos(\alpha) \qquad (1)$$

Vertikala hastigheten:

$$Vy = V \cdot sin(\alpha) \qquad (2)$$

Tid i luften:

$$t = \frac{\left[Vy \quad \sqrt{(Vy^2 + 2 \cdot g \cdot h)}\right]}{g} \qquad (3)$$

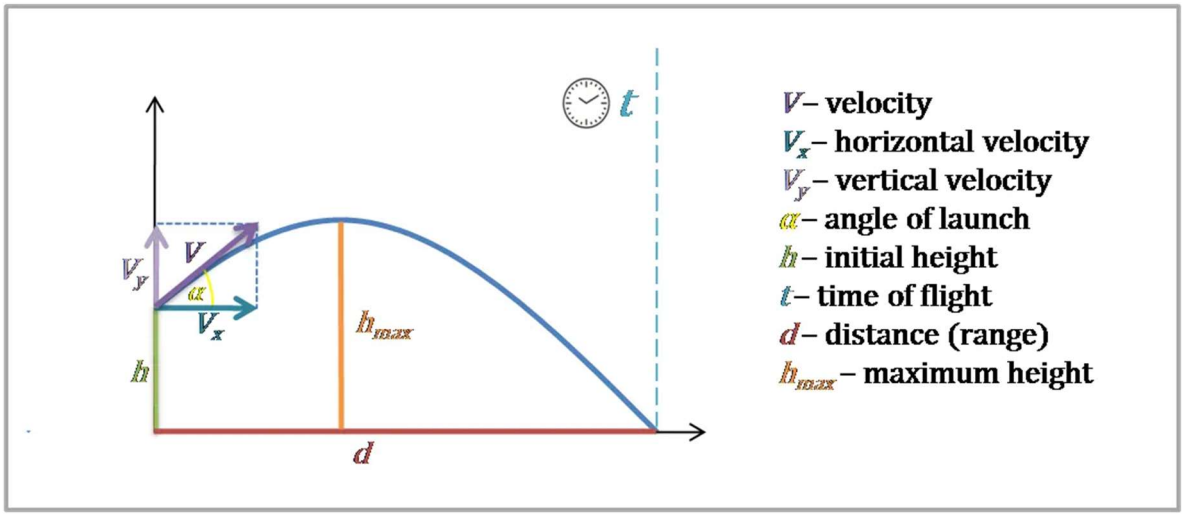
där g är tyngdaccelerationen i Sverige som är ungefär 9,82 m/s² och h är höjden där kastet börjar mätas.

Totala längden av sträckan:

$$d = Vx \cdot \frac{\left[Vy \quad \sqrt{(Vy^2 + 2 \cdot g \cdot h)}\right]}{g} \qquad (4)$$

Kastparabeln räknar ej in luftmotstånd och den enda verkande kraften är gravitationskraften. Figur 2.6 förklarar de olika beteckningarna vidare.

[25]



Figur 2.6: Beskrivning av kastparabeln [26]. Godkänd för användning av källan.

3 Metoder

Detta kapitel presenterar den forskningsmetodik som användes för projektet. Vidare motiveras även de lösningsmetoder som användes för att kunna gå från problemet i avsnitt 1.2 till att uppfylla målen i avsnitt 1.3.

3.1 Forskningsmetodik

De metoder som valdes för detta examensarbete var:

1. Litteraturstudie
2. Utveckling av prototyp för längdberäkning
3. Tester av prototyp
4. Analys av insamlade data

Litteraturstudien gjordes i syfte att bredda och höja kunskapen inom bland annat ämnesområdena artificiell intelligens, maskininlärning och objektdetektering. Dessutom användes studien till att undersöka metoder som skulle kunna användas i projektet. Detta resulterade i flera potentiellt användbara metoder och efter avslutad litteraturstudie togs ett förslag på en lösning fram.

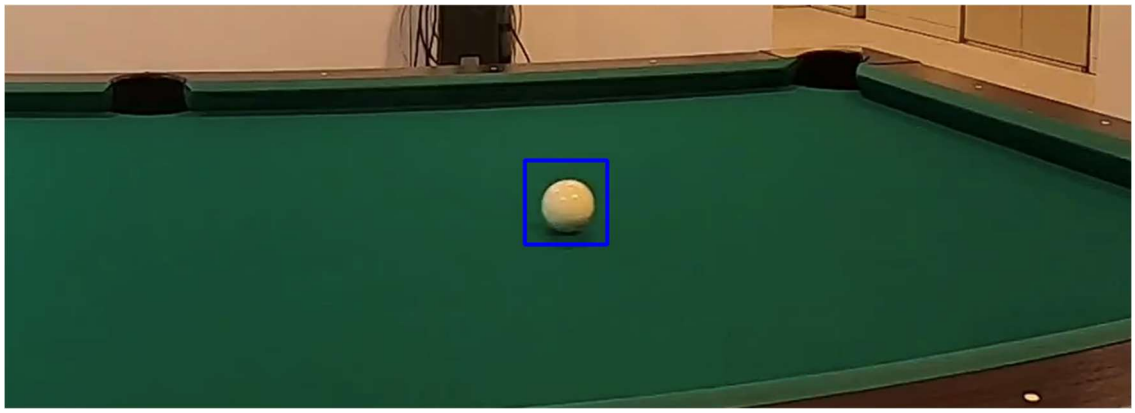
För att utvärdera de valda metoderna och bestämma vilka som passade lösningen bäst användes experiment som metod. Målet med experimentet var att jämföra dessa metoder och fastställa två metoder för objektspårning som kunde användas i det fortsatta arbetet.

Utförandet av experimentet gjordes genom att filma en biljardboll i rörelse som sedan skulle spåras av de olika programvarorna. Detta gjordes inomhus i en kontrollerad miljö utan påverkan av yttre faktorer. Den första metoden som valdes var CSRT, som är en inbyggd funktion i programvarubiblioteket OpenCV. Den andra metoden som valdes var TrackNet, som baseras på artificiell intelligens. Dessa två metoder finns beskrivna i avsnitt 2.3.1 respektive avsnitt 2.5.3.

3.2 Metod CSRT

De tre funktionerna för objektspårning (KCF, TLD, CSRT) som finns beskrivna i avsnitt 2.6 jämfördes genom det experiment som är beskrivet i avsnitt 3.1.

I en körning av programmet tillåts användaren markera det objekt som ska följas med hjälp av en markeringsruta. Markeringsrutan initieras runt objektet, se figur 3.1, och därefter körs videon och spårningsfunktionen försöker därefter följa objektet.



Figur 3.1: Biljardboll vid spårning med CSRT.

Vid jämförelse av de olika metoderna fastställdes det att TLD och KCF inte klarade av att följa biljardbollen och dessa metoder valdes därför bort.

Den metod som fungerade bäst var CSRT då denna metod klarade av att konsekvent följa bollen även under högre hastigheter. Dessutom avbröts spårningen i de fall bollen hamnade ur bild, vilket de två tidigare metoderna inte klarade av.

3.3 Metod TrackNet

Ett av de mål som sattes upp i avsnitt 1.3 var att slutprodukten automatiskt skulle kunna identifiera en kula. Av detta skäl krävdes det en AI till projektet. Att utveckla en helt ny AI ansågs vara för tidskrävande och av detta skäl valdes i stället en redan befintlig metod. Den metod som valdes heter TrackNet och finns beskriven i avsnitt 2.4.1.

TrackNet utvecklades i syfte att följa en tennisboll under en tennismatch. Det var därför inte självklart att metoden skulle fungera i detta projekt. För att fastställa funktionaliteten utfördes ett experiment och återigen användes filmer från en biljardmatch. Resultatet av experimentet var att TrackNet inte kunde följa biljardbollen och av den anledningen drogs slutsatsen att metoden heller inte skulle fungera i kulstötning. Emellertid tack vare att systemet utvecklades med öppen källkod fanns det en möjlighet att bygga ut och utveckla det befintliga programmet och på så vis uppnå denna rapports ändamål. Hur TrackNet utvecklades finns beskrivet i avsnitt 3.4.

En alternativ metod som valdes bort var Toptracer som beskrivs i avsnitt 2.4.2. Den främsta anledningen till att metoden valdes bort var att det är en kommersiell produkt utan öppen källkod. Det hade därför inte varit möjligt att anpassa källkoden för detta projekts syften.

3.4 Utveckling av metoden TrackNet

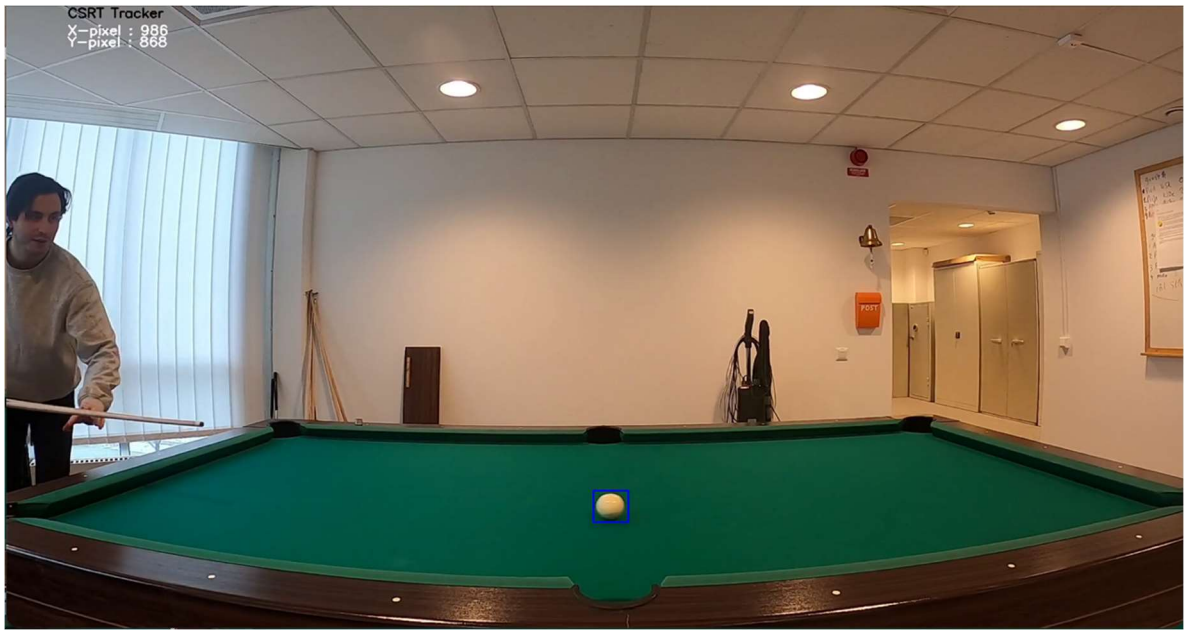
De videosekvenser som skulle användas delades till en början upp i bilder. Varje sekvens var tio sekunder lång och filmades i 60 bilder per sekund, vilket genererade 600 bilder per klipp.

Som indata för upplärning behövde TrackNet tillsammans med bilderna också en lista med data kopplad till varje separat bild. Parametrarna i listan var filnamnet för varje bild, om objektet var synligt eller inte i bilden, x- och y-koordinaten för mittpunkten av objektet som skulle följas. Den fjärde parametern var inte relevant för detta projekt och sattes därför till noll.

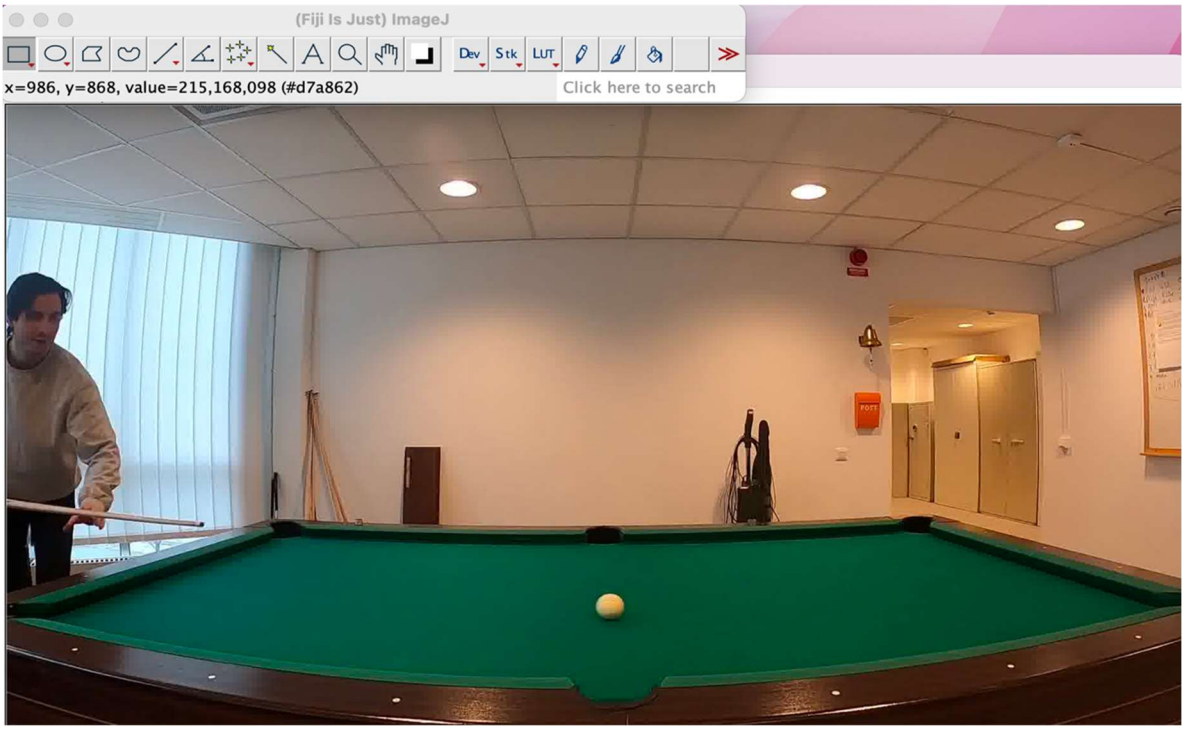
För detta ändamål utvecklades CSRT till att kunna extrahera dessa data. Vid en körning av CSRT sparades dessa parametrar till en CSV-fil, som sedan utgjorde den lista TrackNet använde som indata. Med hjälp av koordinaterna för hörnen på den markeringsruta som är beskriven i avsnitt 3.2 kunde mittpunktens koordinater beräknas.

Därefter säkerställdes att koordinaterna var korrekta genom att jämföra resultatet från CSRT med programmet Fiji ImageJ. Fiji ImageJ är ett bildbehandlingsprogram med öppen källkod som bland annat kan användas för att få information om pixlar i en bild. [27]

Resultatet med CSRT ses i figur 3.2 a), koordinaten för kulans mittpunkt ses i det övre vänstra hörnet, x-koordinaten är 986 och y-koordinaten är 868. Motsvarande resultat i Fiji ImageJ ses i figur 3.2 b) i det övre vänstra hörnet. Eftersom värden med de två olika metoderna överensstämmer antogs beräkningen av koordinaterna vara korrekt.



Figur 3.2, a): Bild från metod med CSRT.



Figur 3.2, b): Bild från Fiji imageJ.

Listan med indata ses i figur 3.3. Den första kolumnen utgör namnet på bilden, den andra kolumnen utgör synligheten som antingen kan vara en etta eller nolla, tredje

och fjärde kolumnen utgör x- respektive y-koordinaten, sista kolumnen är satt till noll för alla rader.

file name	visibility	x-coordinate	y-coordinate	status
000.jpg	1	318	929	0
001.jpg	1	322	926	0
002.jpg	1	328	923	0
003.jpg	1	334	917	0
004.jpg	1	342	911	0
005.jpg	1	350	904	0
006.jpg	1	362	896	0
007.jpg	1	376	884	0
008.jpg	1	392	870	0
009.jpg	1	411	856	0
010.jpg	1	432	844	0

Figur 3.3: CSV-fil innehållande data för TrackNet.

3.5 Datainsamling för TrackNet

Efter att den i avsnitt 3.4 beskrivna metoden för datainsamling säkerställts som tillförlitlig utfördes ett nytt experiment i verklig miljö. Testet ägde rum den fjärde april 2022 på Östermalms idrottsplats. Omgivningen i fälttestet syns i figur 3.4. Experimentet användes dels till insamling av den data som krävdes för att förbättra TrackNet, dels till att utveckla det program som skulle användas för distansberäkning. Nedan följer en beskrivning av experimentets utformning.

På kulstötsbanan placerades tre referenspunkter som refereras till som flaggor i rapporten, dessa ses i figur 3.4. Syftet med flaggorna var att de skulle användas i utformandet av längdberäkningsprogrammet som beskrivs senare i avsnitt 3.6. Med hjälp av dessa var tanken att en relation mellan antalet pixlar och motsvarande höjd eller längd i verkligheten skulle kunna beräknas. Den vänstra flaggan användes i syfte att beräkna relationen mellan höjden och antalet pixlar varför denna flagga också mättes på höjden. De två flaggorna till höger användes i stället i syfte att beräkna relationen mellan längden och antalet pixlar, varför avståndet mellan dessa flaggor mättes och inte höjden. Längden av stöten mättes med hjälp av ett måttband och antecknades. Till vänster i figur 3.4 så syns den gröna kulan som användes. Stötkulan vägde två kg och hade en diameter på tio cm

Testet filmades med kameran GoPro HERO8 med en upplösning på 1920 x 1080 och 60 bilder per sekund. Dessutom, för att få en video med proportionell skala, stängdes vidvinkellinsen av och en linjär lins valdes i stället. 60 bilder per sekund valdes eftersom TrackNet behöver synbar skillnad mellan bilderna för att lära sig att detektera kulan. Det valdes även för att fler bilder per sekund skulle kunna vara för belastande för programmet.

Figur 3.4: Östermalms IP kulstötningsbana.

3.6 Programmet för distansberäkning

Efter valda metoder och datainsamling utvecklades ett längdberäkningsprogram i programmeringsspråket Python. Som tidigare nämnts i avsnitt 2.7 kan längden av stöten beräknas om initiala hastigheten, initiala vinkeln och initiala höjden är känd.

Genom att använda pixelvärdet när kulan befinner sig vid första flaggan samt en virtuell flagga 40 cm till höger kunde två kända positioner fastställas. Utifrån dessa två positioner kunde därefter de trigonometriska definitioner appliceras som syns i figur 3.5, a-c.
För att beräkna den initiala vinkeln användes:

$$tan(\alpha) = \frac{motsatt\ katet}{närliggande\ katet} \quad (5)$$

För att bestämma starthöjden användes den kända höjden av första flaggan. Genom att sätta höjden av flaggan i relation till antalet pixlar kunde sedan höjden på kulan beräknas genom att räkna antalet pixlar mellan kulan och marken.
För den initiala hastigheten användes formeln:

$$hastighet = \frac{sträcka}{tid} \quad (6)$$

Sträckan är densamma som hypotenusan på triangeln i figur 3.5, c).
Formeln som användes för at beräkna hypotenusan var:

$$\cos\alpha = \frac{närliggande\ katet}{hypotenusa} \quad (7)$$

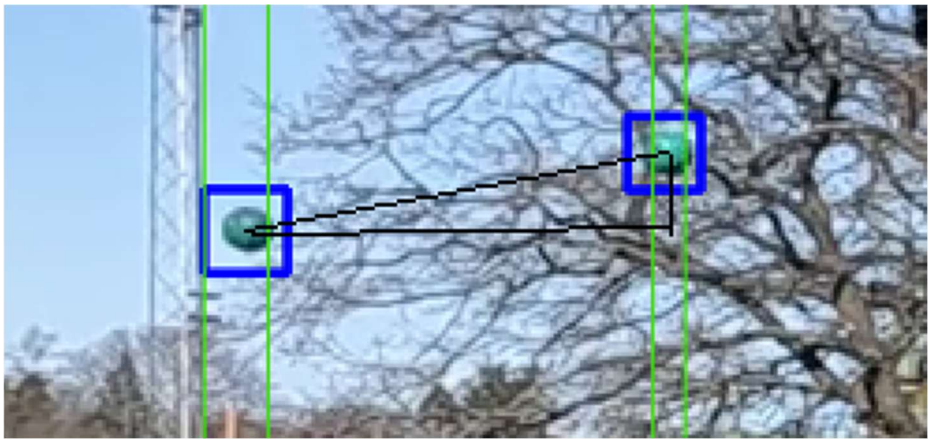
$$Hypotenusa = \frac{närliggande\ katet}{\cos\alpha} \quad (8)$$

För att beräkna den tid som kulan befann sig mellan de två referenserna utnyttjades filmens bildfrekvens som var 60 bilder per sekund. Genom att räkna antalet bilder när kulan befann sig mellan referenserna och sedan dividera resultatet med 60 kunde tiden i sekunder beräknas.

Med initiala vinkeln, initiala hastigheten och initiala höjden kunde sedan en distans beräknas genom att implementera ekvation 4 i programmet för distansberäkning.

Figur 3.5, a): Kulan vid första referenspunkt med CSRT.

Figur 3.5, b): Kulan vid andra referenspunkt med CSRT.



Figur 3.5, c): Kulan vid de två referenspunkterna i relation till varandra med CSRT.

4 Resultat

I detta kapitel redovisas resultatet av längdberäkningen för de två olika metoderna för objektspårning. Dessutom redovisas resultatet av den tränade artificiella intelligensen. Det filmmaterial som användes till träningen av TrackNet och utformandet av längdberäkningsprogrammet finns beskrivet i avsnitt 3.3. I detta kapitel användes ett nytt filmmaterial som samlades in från Upplands Väsby. Varje stöt filmades och längden mättes med hjälp av ett måttband och noterades för referens.

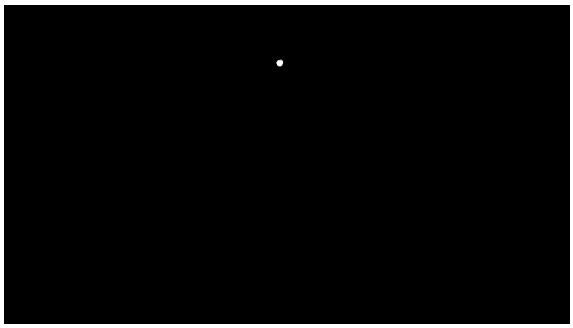
4.1 Resultat upplärning av TrackNet

I figurerna 4.1 ses i den vänstra kolumnen de bilder som programmet fick analysera. Resultatet av dessa bilder ses i kolumnen till höger. Utsignalen för algoritmen blir en svart-vit bild där de delar som uppfattas som en kula markeras vita och övriga delar markeras svarta. TrackNet kunde hitta kulan med 97,88% säkerhet.

Figur 4.1, a): Från Upplands Väsbys stötbana.

Figur 4.1, b): Resultat från TrackNet på figur 4.1, a).

Figur 4.1, c): Från Upplands Väsbys stötbana.



Figur 4.1, d): Resultat från TrackNet på figur 4.1, c).

Figur 4.1, e): Från Upplands Väsbys stötbana.

Figur 4.1, f): Resultat från TrackNet på figur 4.1, e).

4.2 Resultat från manuell mätning

För att jämföra och analysera resultaten av de valda metoderna samlades som tidigare beskrivet data in från en kulstötsbana. I tabell 4.1 presenteras resultaten från tillfället. I den vänstra kolumnen ses antalet stötar och i den högra längden för varje stöt. Längden mättes med hjälp av ett måttband.

Tabell 4.1: Resultat kulstötning, används som referensvärden.

STÖT	LÄNGD (cm)
1.	688
2.	708
3.	739
4.	730
5.	705
6.	716
7.	436
8.	931

4.3 Resultat från metoden CSRT

Tabell 4.2 utgörs av de resultat som erhölls genom att applicera metoden CSRT på filmsekvenserna. På grund av en felkälla i programmet som beskrivs i avsnitt 5.1 applicerades CSRT totalt fem gånger på varje stöt. Detta ses i tabell 4.2 där kolumnerna 1 – 5 utgör varje enskilt resultat från CSRT och den sista kolumnen utgörs av medelvärdet för hela raden.

Tabell 4.2: Varje enskilt resultat med CSRT samt medelvärdet för varje stöt.

STÖT	1 (cm)	2 (cm)	3 (cm)	4 (cm)	5 (cm)	MEDELVÄRDE (cm)
1.	667	680	676	665	668	671
2.	672	674	674	669	668	671
3.	708	711	711	717	711	712
4.	739	737	733	733	739	736
5.	710	706	710	710	698	707
6.	716	701	707	713	707	709
7.	386	378	376	382	379	380
8.	957	957	965	967	948	959

4.4 Resultat från metoden TrackNet

Tabell 4.3 består av de resultat som erhölls genom att applicera metoden TrackNet på filmsekvenserna.

Tabell 4.3: Resultat med TrackNet.

STÖT	LÄNGD (cm)
1.	661
2.	661
3.	741
4.	754
5.	731
6.	717
7.	390
8.	971

5 Analys och diskussion

I detta kapitel analyseras resultatet som presenterades i kapitel 4. Resultatet jämförs med de uppmätta värdena och ställs i relation till projektets målsättning som presenterades i kapitel 1. Vidare diskuteras upptäckta brister och förbättringsmöjligheter samt alternativa lösningsmetoder. Resultatet från TrackNet jämförs sedan med resultatet från CSRT för att bestämma vilken av metoderna som fungerar bäst.

5.1 Analys av resultat från CSRT

Under resultatberäkningen med CSRT noterades det att koordinaterna för kulans centrum varierade beroende på hur noggrant användaren skapade markeringsrutan för kulan. Koordinaterna är en viktig del i programmet för distansberäkning och används för att räkna ut den initiala hastigheten och vinkeln. Dessa variabler används sedan i den matematiska formeln som används för att estimera längden av en stöt. Detta beskrivs i avsnitt 3.5. Följden av detta blev att längduppskattningen av en stöt kunde variera beroende på hur användaren markerade kulan. Av detta skäl gjordes längdberäkning med CSRT fem gånger på varje filmad stöt och det slutgiltiga svaret som ses i tabell 5.1 är medelvärdet av dessa.

Skillnaden mellan CSRT och den riktiga längden ses i kolumnen till höger under rubriken felmarginal. Från tabellen ses att felmarginalen har en variationsbredd på 54 cm med den lägsta felmarginalen i rad fem på två cm och den största felmarginalen i rad sju på 56 cm.

Tabell 5.1: Längdberäkning med CSRT, referenslängden samt felmarginalen.

STÖT	CSRT (cm)	MÅTTBAND (cm)	FELMARGINAL (cm)
1.	671	688	17
2.	671	708	37
3.	712	739	27
4.	736	730	6
5.	707	705	2
6.	709	716	7
7.	380	436	56
8.	959	931	28

Den stora variationsbredden på felmarginalen tros ha flera anledningar. För det första det tidigare nämnda felet med hur kulan markeras i programmet. Detta fel försökte reduceras genom att använda medelvärdet av fem olika resultat.

För det andra upptäcktes det att kameravinkeln hade ändrats mellan varje filmad stöt. På grund av detta kunde inte den korrekta relationen mellan höjden och längden av referenspunkterna och antalet pixlar beräknas. Hur relationen mellan höjden, längden och antalet pixlar beräknas beskrivs i avsnitt 3.5. De olika vinklarna gjorde att den tidigare nämnda relationen inte blev samma för alla stötar. Detta påverkar felmarginalen och gör metoden mindre noggrann.

Från tabell 5.1 ses i stöt fem ett resultat med en felmarginal på två cm. Detta var det enda resultat som uppnådde målet från avsnitt 1.3 att estimera ett resultat inom fem cm. Övriga resultatet med CSRT faller utanför målet. Vidare når CSRT inte heller upp till målet att automatiskt kunna identifiera kulan. Endast målet att kunna följa kulans bana i luften uppnås med hjälp av CSRT.

5.2 Analys av resultat från TrackNet

I tabell 5.2 visas resultaten för kulstötningen samt felmarginalen. Stöt två och stöt sju hade högst felmarginal med 47 cm respektive 46 cm. Stöt sex och tre hade minst felmarginal med 1 cm respektive 2 cm.

Tabell 5.2 Längdberäkning med TrackNet, referenslängd och felmarginalen.

STÖT	TRACKNET (cm)	MÅTTBAND (cm)	FELMARGINAL (cm)
1.	661	688	27
2.	661	708	47
3.	741	739	2
4.	754	730	24
5.	731	705	26
6.	717	716	1
7.	390	436	46
8.	971	931	40

Medelvärdet för felmarginalen blir ungefär 27 cm. Anledningen till att felmarginalen varierar kraftigt vid de olika stötarna är troligen distansberäkningen som beskrivs mer i nästa kapitel.

I de bilder som TrackNet analyserade blev inte kulan helt rund. Detta berodde på att TrackNet upptäckte kulan till 98 %. Detta betyder att när kulnas mittpunkt ska användas för att beräkna längden är det troligt att det inte blir rätt pixelvärde för mittpunkten. Av detta skäl blev resultatet sämre.

Felkällan som orsakades av en varierande kameravinkel som beskrevs i avsnitt 5.1 påverkade även denna metod negativt.

TrackNet klarade två av fyra målsättningar som presenterades i avsnitt 1.3. TrackNet klarade att automatiskt identifiera kulan och att följa kulans bana i luften. TrackNet klarade inte att producera ett mätresultat med en felmarginal på max fem cm eftersom det bara var två av åtta stötar där felmarginalen blev under fem cm. Metoden klarade inte heller att presentera ett mätresultat på mindre än tio sekunder. Anledningen till detta var att inte någon av metoderna har implementerats i realtid.

5.3 Jämförelse av resultat från TrackNet och CSRT

I Tabell 5.3 ses felmarginalen för varje stöt för bägge metoder samt medelvärdet av felmarginalerna.

Tabell 5.3: Felmarginal för TrackNet och CSRT.

STÖT	TRACKNET FELMARGINAL (cm)	CSRT FELMARGINAL (cm)
1.	27	17
2.	47	37
3.	2	27
4.	24	6
5.	26	2
6.	1	7
7.	46	56
8.	40	28
Felmarginal medelvärde	27	22

Vid jämförelsen av resultaten visas det att CSRT hade en lägre felmarginal än TrackNet, även om marginalen inte var särskilt stor. Eftersom bägge metoder använder sig av samma formler för att beräkna längden blir det pixelvärdet för kulans centrum som skiljer sig för metoderna. Hur mittpunkten blir bestämd i metoderna beskrivs i kapitel 3.

Dessutom blir den största felkällan formeln som används vid beräkningen av längden. Formeln utgår från att kastet är rakt vilket det inte alltid är i kulstötning och

vissa av resultaten i rapporten reflekterar detta. Stötarna som är sneda har större felmarginaler än de som är rakare.

En nackdel med metoderna är att de bägge är programmerade för ett visst avstånd mellan kamera och stötbana. Om programmet skulle användas vid en annan kulstötning där kameran och stötbanans avstånd skiljer sig från det nuvarande, skulle programmet behöva programmeras om för att få ett resultat.

Även om CSRT hade en mindre felmarginal än TrackNet pekar ändå resultaten på att TrackNet är den bättre metoden. Detta beror på att TrackNet klarade två av fyra målsättningar medan CSRT bara klarade en. Eftersom TrackNet identifierar kulan automatiskt finns det även potential att implementera systemet i realtid. TrackNet skulle även kunna vidareutvecklas för att få ett noggrannare pixelvärde för mittpunkten av kulan och få ett mer exakt resultat.

5.4 Alternativa lösningar

Prototypen uppfyller i dagsläget inte de mål som sattes upp i kapitel 1 men utifrån resultatet finns en stor möjlighet att förbättra systemet. Författarna bakom rapporten har identifierat några alternativa lösningar och förbättringar som kan vara användbara vid framtida arbeten.

I projektet har all den data som inhämtats för träning av TrackNet utgjorts av data författarna själva producerat. På grund av tidsbrist är mängden träningsdata begränsad till vad som hann spelas in under en förmiddag. Av detta skäl har systemet för automatisk identifiering och objektspårning inte kunnat tränas på den mängd data som egentligen var nödvändig. En alternativ lösning till datainhämtningen skulle därför kunna vara att undersöka möjligheten att använda tidigare inspelningar, till exempel gamla tevesändningar från tävlingar eller liknande evenemang.

Som en följd av mer träningsdata skulle också en alternativ lösning kunna vara att från grunden utveckla en artificiell intelligens. Fördelarna med detta skulle vara en objektspårning med högre precision vilket skulle resultera i ett noggrannare mätresultat. Dessutom skulle ett sådant system vara mer tillförlitligt och fungera under fler förutsättningar än det nuvarande. Detta vore bra eftersom den framtagna prototypen är utformad och testad under soliga väderförhållanden och därför inte med säkerhet fungerar i andra väderlekar eller andra miljöförhållanden.

Metoden Toptracer som beskrevs i kapitel 2 är även en lösning som skulle kunna utforskas vidare. Eftersom denna metod är utvecklad för golf krävs stor justering men teorin bakom metoden är ändå lovande. Toptracer har dock ingen öppen källkod och kräver avancerad utrustning.

Det finns även en chans att TrackNet hade haft större precision om träningsdata hade registrerats manuellt i stället för med CSRT. Med manuellt menas att studera bild för bild och markera pixelvärdet för kulan, synligheten och status. Beroende på antalet filmer som ska bearbetas kan detta dock vara ett alltför tidskrävande projekt.

5.5 Fördelar med programmet i ett bredare perspektiv

Även om systemet i dagsläget inte uppfyller kraven på noggrannhet och precision från kapitel 1 ses flera fördelar och tillämpningsområden med nuvarande prototyp.

Systemet kan dels användas i andra sammanhang där kraven på noggrannhet är lägre än vid tävling. Ett exempel skulle kunna vara vid träningstillfällen, där det kanske är av större vikt att få ett snabbt men mindre precist svar. Dessutom då längdberäkningen sker automatiskt behöver ingen tränare eller funktionär befinna sig på kulstötsbana. Således hade också risken för personskador minskat. Vad mera är att det från programmet går att utvinna ytterligare data, till exempel den initiala vinkeln och hastigheten av stöten. Dessa parametrar skulle kunna vara intressanta för utövare av sporten som vill förbättra sina resultat.

Om nuvarande prototyp utvecklades till att ge ett mer precist svar hade den kunnat implementeras i tävlingssammanhang. Förutom att reducera risken för skador hade detta även kunnat resultera i färre felaktiga mätningar vilket i sin tur hade lett till ett ökat förtroende till resultatet. Möjligtvis skulle också ett ökat förtroende kunna leda till en ökad popularitet för sporten. Slutligen finns det också en ekonomisk fördel i att använda ett automatiskt system för mätning i stället för att anställa personal för den uppgiften.

6 Slutsats

Denna rapport har med hjälp av två olika metoder utvecklat en prototyp för längdberäkning inom kastgrenen kulstötning. Den ena metoden har baserats på artificiell intelligens (TrackNet) och den andra metoden var en inbyggd funktion för objektspårning (CSRT). Resultatet från de båda metoderna jämfördes sedan i syfte att bestämma vilken av de båda metoderna som bäst uppnådde rapportens mål.

TrackNet uppnådde två av de fyra målen som definierades i projektet medan CSRT endast uppnådde ett av målen. TrackNet klarade att automatiskt identifiera kulan samt följa kulans projektilbana medan CSRT bara kunde följa kulans projektilbana. De målen som inte uppnåddes var att presentera ett mätresultat inom tio sekunder och estimera längden med en felmarginal på maximalt fem cm.

Den nuvarande prototypen uppfyller inte de krav på noggrannhet som hade krävts vid tävlingssammanhang. Däremot ser rapportens författare en potential i att använda den i sammanhang med lägre ställda krav. Det skulle till exempel kunna vara vid träning där det är viktigare att få ett snabbt men ungefärligt resultat. Därtill skulle risken för skador reduceras samt möjliggöra att själv kunna bedriva träning då det inte skulle behövas någon ytterligare person för själva mätningen.

Författarna till rapporten föreslår till framtida arbeten att:

- Filma från fler vinklar för att kunna beräkna sneda kast mer noggrant.
- Undersöka hur luftmotståndet påverkar stötens längd.
- Utveckla objektdetekteringen till att innefatta även andra kastgrenar, till exempel spjut, slägga eller diskus.

Källförteckning

[1] World Athletics. Javelin Throw Men [Internet]. Monaco: World Athletics; 2022 [citerad 2022-06-08]. Hämtad från: https://www.worldathletics.org/records/all-time-toplists/throws/javelin-throw/outdoor/men/senior

[2] Johnsson, P. Se skandalmätningen i kula [Internet]. Stockholm: Svt Sport; 2017 [uppdaterad 2017-08-11; citerad 2022-04-26]. Hämtad från: https://www.svt.se/sport/friidrotts-vm/se-skandalmatningen-i-kula

[3] Wennerholm, M. Dödad – av en kula i huvudet [Internet]. Stockholm: Aftonbladet; 2005 [uppdaterad 2011-03-08; citerad 2022-04-26] Hämtad från: https://www.aftonbladet.se/sportbladet/a/gPwdrJ/dodad--av-en-kula-i-huvudet

[4] Berkeley. What is machine learning (ML)? [Internet]. California: Berkeley School of Information; 2020 [Citerad 2022-04-25]. Hämtad från: https://ischoolonline.berkeley.edu/blog/what-is-machine-learning/

[5] IBM Cloud Education. Artificial Intelligence (AI) [Internet]. IBM; 2020 [Citerad 2022-04-25]. Hämtad från: https://www.ibm.com/cloud/learn/what-is-artificial-intelligence

[6] IBM. What is computer vision? [Internet]. IBM; 2022 [Citerad 2022-04-26] Hämtad från: https://www.ibm.com/topics/computer-vision

[7] Mihajlovic, I. Everything You Ever Wanted to Know About Computer Vision. Here's A Look Why It's So Awesome. [Internet]. Towards Data Science; 2022 [uppdaterad 2019-04-25; Citerad 2022-04-26]. Hämtad från: https://towardsdatascience.com/everything-you-ever-wanted-to-know-about-computer-vision-heres-a-look-why-it-s-so-awesome-e8a58dfb641e

[8] Fritz AI. Object Detection Guide. [Bild] Fritz AI; 2022 [Citerad 2022-04-26] Hämtad från: https://www.fritz.ai/object-detection/

[9] IBM Cloud Education. Convolutional Neural Networks [Internet]. IBM; 2020 [Citerad 2022-04-25]. Hämtad från: https://www.ibm.com/cloud/learn/convolutional-neural-networks

[10] Huynh N. Convolutional Neural Network (CNN) overview [Internet] medium; 2018 [Citerad 2022-05-25]. Hämtad från: https://medium.com/@eyeq/convolutional-neural-network-cnn-overview-33026f15dd28

[11] Stanford. Convolution neural networks (cnns / convnets) [Internet]. California: CS231n; 2022 [Citerad 2022-04-25] Hämtad från: https://cs231n.github.io/convolutional-networks/

[12] Saha S. A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way [Bild]. Medium; 2018 [Citerad 2022-04-25]. Hämtad från: https://towards-datascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53onal-neural-networks-the-eli5-way-3bd2b1164a53

[13] S. Raschka, V. Mirjalili, “Python machine learning: Machine learning and deep learning with python,” *Scikit-Learn, and TensorFlow. Second edition ed*, 2017

[14] L. G. Hamey. A functional approach to border handling in image processing [Internet]. DICTA; 2015 [Citerad 2022-04-25]. Hämtad från https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7371214

[15] Yamashita, R., Nishio, M., Do, R.K.G. et al. Convolutional neural networks: an overview and application in radiology [Bild]. Insights into Imaging; 2018 9, 611–629 [Citerad 2022-04-24] Hämtad från: https://doi.org/10.1007/s13244-018-0639-9

[16] Huang YC, Liao IN, Chen CH, İk TU, Peng WC. TrackNet: A deep learning network for tracking high-speed and tiny objects in sports applications [Internet]. IEEE; 2019 16EE. [Citerad 2022-04-25]. Hämtad från: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8909871

[17] Sung S. Track a high-speed moving object with TrackNet [Internet]. Medium; 2021 [Citerad 2022-04-25]. Hämtad från: https://medium.com/geekculture/track-a-tennis-ball-with-computer-vision-4f8d2f9c0412

[18] McQuade M. Toptracer changes way fans watch, practice sport of golf [Internet]. Arizona: Cronkite News - Arizona PBS; 2020 [Citerad 2022-04-25]. Hämtad från: https://cronkitenews.azpbs.org/2020/07/31/toptracer-golf-technology/

[19] Ryan S. Was Collin Morikawa's drive on 16 the greatest shot in PGA Championship History? [Bild]. California: GolfDigest; 2020 [Citerad 2022-04-25]. Hämtad från: https://www.golfdigest.com/story/pga-championship-2020-was-collin-morikawa-16th-hole-drive-tpc-harding-park-the-greatest-in-pga-history

[20] Python Software Foundation. Python Software Foundation [Internet]. Delaware: Python; 2022 [Citerad 2022-04-25] Hämtad från: https://www.python.org/psf/

[21] Business Insider India. Difference between Compiler and Interpreter [Internet]. Indien: Business Insider India; 2022 [Uppdaterad: 2019-05-27; Citerad: 2022-04-25] Hämtad från: https://www.businessinsider.in/difference-between-compiler-and-interpreter/articleshow/69523408.cms

[22] Woodie, A. What’s driving python’s massive popularity[Internet]. Datanami; 2021. [Citerad 2022-05-16]. Hämtad från: https://www.datanami.com/2021/10/20/whats-driving-pythons-massive-popular

[23] Open source Initiative. The open source definition [Internet]. Open source Initiative; 2007 [Uppdaterad: 2007-03-22; Citerad: 2022-04-25] Hämtad från: https://opensource.org/osd

[24] Kulhary R. OpenCV - Overview [Internet]. GeeksforGeeks; 2021 [Citerad 2022-04-25]. Hämtad från: https://www.geeksforgeeks.org/opencv-overview/

[25] Rabinovitch M. Comparing state of the art region of interest trackers [Internet]. Medium; 2019 [Citerad 2022-04-25]. Hämtad från: https://medium.com/teleidoscope/comparing-state-of-the-art-region-of-interest-trackers-906ba420e80d

[26] Szyk B, Pamuła H. Projectile Motion Calculator [Internet]. Omni Calculator; 2021 [Citerad 2022-04-25]. Hämtad från: https://www.omnicalculator.com/physics/projectile-motion

[27] Rasband, W. Introduction [Internet]. Maryland: ImageJ [Citerad 2022-06-08]. Hämtad från: https://imagej.nih.gov/ij/docs/intro.html

TRITA – CBH-GRU-2022:079

www.kth.se