CHALMERS

E-blackboard

Från krittavla till webbgränssnitt

Kandidatarbete inom Data- och Informationsteknik

Simon Andersson
Saudin Botonjic
Christoffer Lundgren
Dennis Parviainen
Tomas Selldén

Chalmers tekniska högskola
Göteborgs universitet
Institutionen för Data- och Informationsteknik
Göteborg, Sverige, Juni 2014

E-blackboard

Simon Andersson
Saudin Botonjic
Christoffer Lundgren
Dennis Parviainen
Tomas Selldén

Examiner: Arne Linde

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden June 2013

Förord

Vi vill tacka flera personer som har hjälpt oss under projektets gång. Ett stort tack till E-sektionens Teletekniska Avdelning för praktiska råd och konstruktionsmaterial. Tack till vår handledare, universitetslektor på avdelningen för Datorteknik, Lena Peterson. Lena har visat tillit samt givit kontinuerlig feedback genom hela projektet.

Tack även till lektor på avdelningen för Datorteknik, Arne Linde för rådgivning och godkännandet av inköpen som gjorde projektet möjligt att genomföra. Övriga tack går till Tord Hansson och Birger Signal, gruppledare respektive tekniker på AV- och lokalservice på Chalmers Tekniska Högskola, för råd vid installation samt montering.

Göteborg Juni 2014

Saudin, Tomas, Simon, Christoffer och Dennis

Sammanfattning

Denna kandidatrapport behandlar utvecklingen av ett semi-autonomt system som hjälper studenter genom att elektroniskt tillhandahålla information som presenteras på krittavlor under en föreläsning. Systemet använder sig av enkortsdatorn Raspberry Pi med tillhörande kameramodul monterad på ett roterande stativ för att kunna fotografera samtliga krittavlor i föreläsningssalen. Stativets rotation styrs från Raspberryn via en servomotor. Bilder tagna av systemet redigeras automatiskt på Raspberryn med hjälp av grafikbiblioteket SimpleCV i ett pythonscript. Bilderna publiceras på ett webbgränssnitt där studenter kan ta del av dem och repetera det som behandlades under föreläsningen.

Projektet resulterade i en prototyp där en bild publiceras på webbgränssnittet cirka 40 sekunder efter att en tavla skrivits färdigt. Denna fördröjning anses godtagbar då en tavla ytterst sällan behöver dokumenteras med kortare intervall. Den automatiska bildredigeringen korrigerar perspektivet efter behov och klipper ut tavlan så att ingen onödig information publiceras på webbgränssnittet. Systemet kräver minimal interaktion från föreläsare i form av en sensoraktivering som sker då en tavla skjuts upp till tavelställningens översta läge. Det övergripande resultatet av detta projekt indikerar att ett system med denna funktionalitet är realiserbart och skulle troligtvis ha en positiv inverkan på studenters inlärning.

Abstract

This bachelor thesis covers the development of a semi-autonomous system which helps students by electronically providing information presented on blackboards during a lecture. The system consists of a camera module connected to a Raspberry Pi mounted on a rotating frame, which enables the system to cover all of the boards in the lecture hall. The frame's rotation is controlled by a servo-engine connected to the Raspberry Pi. Images captured by the system are automatically processed on the Raspberry Pi through the use of the graphics framework SimpleCV in a python script. The images are published on an online interface where students can access them and rehearse the content covered during the lecture.

The project ultimately resulted in a prototype where an image is presented on the online interface approximately 40 seconds after a blackboard has been completed. This delay is considered acceptable since a board rarely has to be documented more frequently. The automatic image processing will correct the perspective as needed and crop the image to exclude any unnecessary information before publishing the image on the online interface. The system requires minimal interaction from the lecturer. This consists of moving the blackboard to the highest position of the board's frame. The overall result of this project indicates that a system with this type of functionality can be implemented and is likely to have a positive impact on students' learning.

Ordlista

Agil systemutveckling Samlingsnamn för projektmodeller som fokuserar på flexibilitet.

Blob Ett område i en bild som avgränsats av olika egenskaper, vanligtvis skillnader i färg- och ljusstyrka.

Center Stage Designmönster som beskriver hur ett gränssnitt bör designas då dess primära uppgift är att belysa en specifik funktion.

Datorseende Forskningsfält inom bearbetning och analys av bilder.

Detektor Elektrisk komponent vars funktion är att ta emot signaler.

Emitter Elektrisk komponent vars funktion är att sända ut signaler.

Escape Hatch Designmönster som beskriver hur ett gränssnitt bör designas då sidorna utgör någon typ av process .

GIT Versionshanteringsprogram som sparar samtliga versioner av till exempel kodfiler eller textdokument för revidering eller sammanslagning.

GPIO General Purpose Input Output är de ingångar/utgångar, på integrerade kretsar, vars beteende är programmeringsbart av användaren.

Hough-transform En transform som används inom datorseende for att identifiera specifika egenskaper i en bild.

Interaktiv whiteboard Tryckkänslig whiteboard som kopplas till en dator för digitalisering av det skrivna innehållet.

One-Window Drilldown Designmönster som beskriver hur ett gränssnitt bör designas då det finns begränsat med utrymme eller då den intressanta informationen tar stor plats.

Potentiometer Ett variabelt elektriskt motstånd.

Scrum En agil systemutvecklingsmetod där arbetet delas upp i iterationer med bestämda tidsperioder.

Sprint En iteration i scrum-modellen.

Translatera Inom matematiken, en förflyttning av koordinater enligt ett förbestämt mönster.

Innehållsförteckning

1

Inledning

I följande kapitel ges en introduktion till projektet. I korthet behandlas varför projektet är aktuellt, vilka alternativa lösningar som undersökts, hur det slutgiltiga systemets ingående delar arbetar i relation till varandra och de avgränsningar som gjorts.

1.1 Bakgrund

Ett dilemma som är vanligt förekommande hos studenter är huruvida de ska anteckna allt som föreläsaren skriver på krittavlan eller om de ska lägga fokus på att lyssna och förstå. Flera väljer det senare alternativet, då det är krävande att ta in den information som presenteras samtidigt som den måste skrivas ned och risken finns att man i slutändan sitter med oförståeliga anteckningar. Väljer man att enbart lyssna under föreläsningen kan det vara omöjligt att få tag i föreläsningsanteckningar i efterhand.

I dagsläget finns det olika tekniker för att underlätta studenternas inlärning: exempelvis powerpointpresentationer och interaktiva whiteboards. Båda teknikerna har dock sina nackdelar. En powerpointpresentation är ofullständig om föreläsaren skriver information på tavlorna parallellt med presentationen. Interaktiva whiteboards möjliggör digitalisering av föreläsningsanteckningar men har nackdelen att de är dyra.

1.2 Syfte

Projektets syfte var att underlätta studenters studier genom att utveckla ett system som elektroniskt tillhandahåller information som presenteras på krittavlor utan att studenter behöver anteckna. Vidare ska systemet i största möjliga utsträckning arbeta autonomt med minimal interaktion från användare och administratörer.

1.3 Förstudie

För att bestämma vilken lösningsmetod som bäst appliceras på projektets syfte gjordes en undersökning av olika lösningsförslag. Tre potentiella tekniker undersöktes: en skanner som läser av tavlan då en knapp eller en sensor aktiverats, att använda Microsofts Kinect för att identifiera och ta en bild på tavlan, och en digitalkamera som tar en bild på en tavla då en knapp eller en sensor aktiveras.

Skanning är en teknik som läser av ett analogt format och digitaliserar det för vidare digital användning. Idag används den främst för inläsning av dokument. Den vanligaste typen är planskannern där objektet som ska skannas läggs på en glasyta och med hjälp av ett rutnät delas upp i avläsningspunkter som sedan sparas som pixlar [1]. Digitalisering av ett analogt format sker genom att skannern skickar vitt ljus som reflekteras mot objektet och kommer tillbaka till skannern med data om kulören för varje punkt. Efter att data har samlats in från hela dokumentet så sätts en digital bild ihop. Tekniken ställer höga krav på omgivningen; dels en mörk miljö utan störningar och dels att skannern ligger mycket nära objektet. Dessa krav på miljön innebär att tekniken inte är optimal vid användning på en krittavla och således förkastades förslaget.

Microsoft Kinect är en avancerad kamera som är utvecklad för att känna av rörelser i en 3D-miljö och utvecklades för att styra Microsofts egna spelkonsol Xbox 360. Kinect är uppbyggd av en projektor som skickar ut infrarött ljus samt två kameror: en RGB- och en infraröd kamera. Med hjälp av projektorn och de två kameror får Kinect-kameran en uppfattning om var saker finns i rummet. Då kameran främst fokuserar på att känna av skillnader i djup kan kameran inte direkt utnyttjas för att hitta en färdigskriven krittavla [2]. Trots att kameran kan användas för att ta stillbilder förkastades denna idé, då den är dyr i jämförelse med andra kameror som kan ta likvärdiga bilder [3].

Om en digitalkamera ska användas krävs ytterligare komponenter för att systemet ska uppnå önskad grad av automation. Det behöver finnas en sensor som signalerar till kameran när en bild ska tas. När det väl finns en bild måste den skickas

till ett system för vidare bearbetning. Idag finns minneskort med trådlös funktionalitet som medför en möjlighet att använda en vanlig digitalkamera för att automatiskt ladda upp bilder på internet [4]. Lösningsmetoden med digitalkamera är genomförbar men förkastades på grund av att det var för dyrt.

Ett annat alternativ som följer samma princip är att använda en kamera i kombination med en enkortsdator som både kan ta bilder och utföra beräkningar på en och samma plats. Raspberry Pi är ett exempel på en enkortsdator med möjlighet för anslutning av både sensor och kameramodul. Då kostnaden av både Raspberry Pi och tillhörande komponenter är låg (se bilaga A) valdes denna lösningsmetod för implementering av systemet. Valet medför att interaktion från en föreläsare behövs för att systemet ska kunna veta när en tavla är färdigskriven.

1.4 Problemformulering

Efter att vi formulerat projektets syfte och förstudien var genomförd, skapade vi projektets problemformulering. Vi bröt ned projektet till fyra delproblem: övergång från den analoga krittavlan till ett digitalt format, kvalitetssäkring av bilden, publikation av informationen på ett webbgränssnitt samt att systemet inte påverkar eller stör under en föreläsning.

För att produkten ska bli användbar behöver systemet kunna garantera att all information som föreläsaren skriver på en krittavla finns tillgängligt för slutanvändaren.

För att undvika att webbgränssnittet presenterar bilder som är ofullständiga eller av bristande kvalité bör en bildanalys implementeras i systemet. Vidare bör tavlan vara det enda som syns på bilden för att underlätta läsning av tavlan och undvika att föreläsaren visas på gränssnittet.

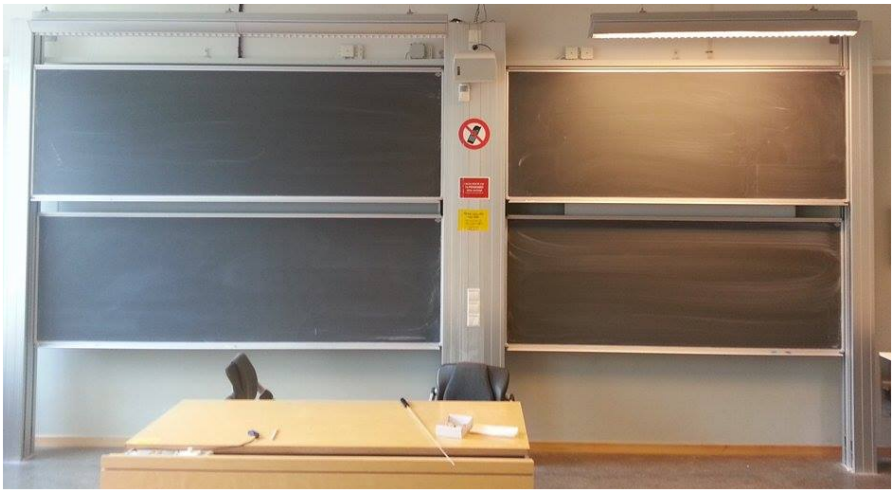
Om studenter ska använda systemet är det väsentligt att webbgränssnittet är användarvänligt. Gränssnittets huvuduppgift är att förse användaren med specifika anteckningar på ett enkelt och effektivt sätt. Den information som presenteras ska vara lättöverskådlig för att användaren snabbt ska hitta det som söks.

Slutligen är det viktigt att systemet inte har någon större inverkan på hur föreläsningar läggs upp. Därmed ska systemet fungera autonomt efter att en sensor blivit aktiverad, samt att hela processen inte tar för lång tid. För att inte missa aktiveringar av sensorn bör hela processen aldrig ta längre tid än tre minuter.

1.5 Systemöversikt

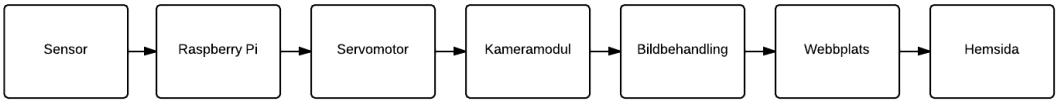
Systemet består av sensorer, Raspberry Pi, Raspberry Pi kameramodul, servomotor och en webbplats innehållandes en databas och ett webbgränssnitt.

Systemet har monterats i en specifik föreläsningssal, se figur 1.1. Varje tavla har en egen sensor som är fäst högst upp på tavlans ställning. Vidare är Raspberry Pi med tillhörande kameramodul och servomotor fäst i taket fyra meter från krittavlorna.



Figur 1.1: *Specifik föreläsningssal för systemet.*

Figur 1.2 visar hur systemets komponenter arbetar med varandra. När en tavla skjuts upp aktiveras en sensor som skickar en signal till Raspberry Pi:n. Vidare skickas signaler till en servomotor som riktar in kameramodulen mot den tavla vars sensor skickat signal. Därefter tar kameran ett kort på tavlan som sedan processeras för att säkerställa att bilden är läsbar och göra justeringar för att förbättra bildkvaliteten. Bilden tillsammans med tillhörande metadata skickas därefter till webbplatsen och presenteras på gränssnittet ungefär 40 sekunder efter att bilden tagits.



Figur 1.2: *Förenklat blockschema över E-blackboard systemet med huvudkomponenterna utritade.*

1.6 Metod

Som första steg i arbetet genomfördes en litteraturstudie med målet att ta fram den bästa lösningen till problemet att digitalisera föreläsningsanteckningar. Både böcker och internet användes flitigt för att få en uppfattning om hur olika tekniker fungerade.

Programmet som körs på Raspberry Pi:n är skrivet i Python med vissa inslag av andra språk då enheten ska interagera med andra delar av systemet. Vidare användes SQL, PHP, HTML, CSS och Javascript vid utveckling av webbgränssnittet. Språken valdes baserat på tidigare erfarenheter. För att möjliggöra parallell utveckling från flera arbetsstationer har versionshanteringsprogrammet GIT [5] använts. GIT sparar dessutom historik, vilket gör det möjligt att gå tillbaka i historiken när problem uppkommer. Matlab har använts för beräkningar och plottar.

En agil systemutveckling [6] i form av scrum [7] har använts under projektets gång. Detta innebär att arbetet delats upp i olika sprintar där varje sprint har innehållit ett visst antal uppgifter och pågått en bestämd tid och avslutats med testning och utvärdering. Programmet Trello [8] har använts för hantering av sprintar. Programmet finns både som webbgränssnitt och som mobil applikation, vilket underlättat arbetet avsevärt.

1.7 Avgränsningar

Den sal som systemet testats i är EDIT husets EA-sal. Denna sal valdes till följd av att Chalmers gav oss tillåtelse att montera upp ett stativ för Raspberry Pi:n i denna sals tak. En annan avgränsning som gjorts är att inte särbehandla de tavlor som rättats och eventuellt läses in en andra gång. Detta skulle kräva att varje ny tavla jämförs med alla tidigare inlästa tavlor från samma föreläsning. Att implementera en sådan funktion skulle ta tid från andra delområden som har prioriterats högre. Under projektet har det inte skett någon undersökning för att ta reda på vem som äger innehållet som presenteras på en krittavla under en föreläsning och hurvida det är tillåtet att publicera detta innehåll för allmänheten. Det har inte heller skett någon utveckling av egna algoritmer för bildbehandling. Projektet fokuserar att utveckla ett komplett system och det ansågs allt för resurskrävande att utveckla dessa algoritmer från grunden.

2

Teori

I detta kapitel beskrivs systemets olika komponenter och hur respektive komponent fungerar.

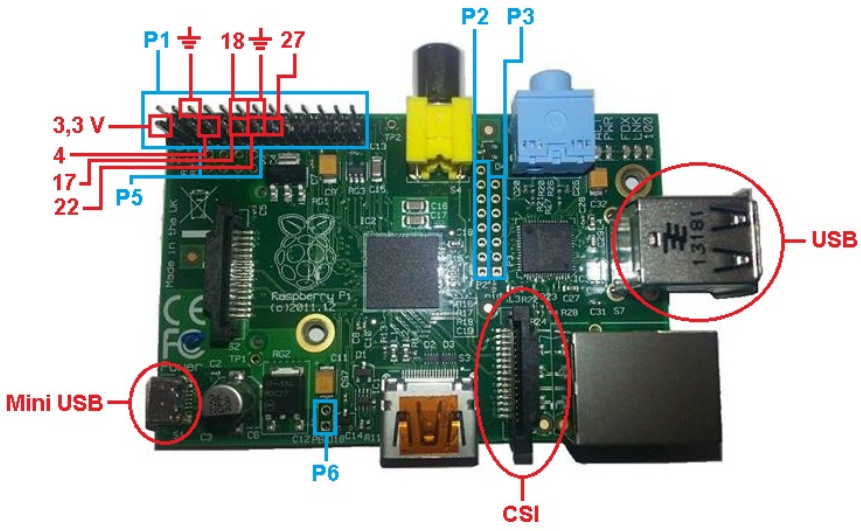
2.1 Raspberry Pi

Raspberry Pi är ett datorsystem som ingår i kategorin enkortsdatorer [9], vilket innebär att samtliga komponenter sitter på ett enda kretskort. Datorsystemet är därmed inte större än en handflata. Prestandan blir dock begränsad då det inte finns plats för kraftfullare hårdvara. Raspberry Pi finns i två modeller och skillnaden är huvudsakligen prestandan och vilka I/O-portar som finns tillgängliga [10] [11]. Raspberry Pi är byggd för att köra Linux. Det vanligaste operativsystemet heter Raspbian [12] och är baserat på linuxdistributionen Debian.

Raspberry Pi har fem olika uttag, som visas i figur 2.1 tillsammans med andra relevanta delar. Uttaget som benämns P1 består av 26 stycken 2,54 mm hanestift varav två matar 3,3 V och två andra matar 5 V. Dessa fyra stift är konstant spänningsmatande efter systemets uppstart. Av de 22 kvarvarande stiften är fem jordade och resterande är GPIO-stift, *General Purpose Input Output*. GPIO-stiften är programmerbara och kan användas för att antingen ta emot eller skicka signaler. När stiften används för att skicka en signal kommer denna utsignal att ha en amplitud på 3,3 V [13].

Uttagen som benämns P2 och P3 används enbart vid produktion. Uttaget P5 kan ses som en utökning av P1 med ytterligare spänningsmatande stift, jord och GPIO-stift. Det sista uttaget, P6, består endast av två stift där det ena startar om processorn då det kopplas till jord och det andra stiftet är jord.

Slutligen finns det ett CSI-uttag, *Camera Serial Interface*, som är anpassat för Raspberry Pi:ns egna kameramodul.



Figur 2.1: *Raspberry Pi model B med de markerade uttag som används i projektet.*

Kameramodulen [14] är en högupplöst kamera med funktionalitet för att fånga både film och stillbild. Maxupplösning för filmtagning är 1920 x 1080 pixlar och stillbildstagning 2592 x 1944 pixlar. Modulen ansluts till Raspberry Pi:s CSI-uttag och styrs av specialiserad programvara.

2.2 Sensorer

En sensor är en komponent vars uppgift är att upptäcka förändringar i miljön. Då en förändring sker skickas information generellt sett till ett annat system som reagerar därefter [15]. Olika miljöer och situationer kräver olika typer av sensorer och nedan beskrivs tre typer som övervägdes för projektet.

En mekanisk brytare är en öppen krets som sluts då tillräcklig kraft appliceras på brytaren och en signal går då fram. Storleken på den kraft som krävs för att

sluta kretsen bestäms av en inbyggd fjäder. Liknande mekaniska brytare är de som används i mekaniska tangentbord [16].

Ett annat alternativ är IR-sensorn [17]. Den består av en emitter och en detektor. Emittern är en diod som skickar ut infrarött ljus och detektor avger en elektrisk impuls då den träffas av infrarött ljus. IR-sensorn behöver anpassas för den miljö den ska installeras i, då ett relativt tröskelvärde måste anges i form av kvoten mellan utskickat och uppfångat infrarött ljus. När miljön är mörkare sätts tröskelvärdet lägre och då miljön är ljusare sätts tröskelvärdet högre.

Halleffektsensorn [18] regerar på magnetfält och avger en spänning då ett magnetfält uppträder vinkelrätt mot sensorn. Generellt sett är den avgivna spänningen relativt svag och sensorn brukar därför användas tillsammans med likströmsförstärkare och spänningsregulatorer.

2.3 Webbplats

I följande kapitel beskrivs de delar som ingår i en webbplats. De vanligaste typerna av webbservrar presenteras följt av hur webbgränssnitt är uppbyggda. Detta följs av en beskrivning av olika databaser och deras tillämpningsområden samt en redogörelse av protokollet SSH.

2.3.1 Webbserver

För att publicera ett webbgränssnitt krävs en webbserver, en kombination av hårdvara och mjukvara som är konfigurerad för publik åtkomst till gränssnittet. Det finns flera olika sätt att tillhandahålla en webbserver, varav de vanligaste innefattar följande: ett webbhotell, en VPS, *Virtuell Privat Server*, och en dedikerad server.

Det enklaste sättet att tillhandahålla en webbserver är att hyra ett webbhotell. Ett webbhotell innebär att en server delas av flera användare, vilket leder till att konfigurering av webbservern är högst begränsad. Istället får användare anpassa sig till de funktioner som webbhotellet erbjuder.

En VPS [19] är en virtuell server som huseras på en fysisk server tillsammans med andra virtuella servrar. Denna teknik ger användare tillgång till att konfigurera webbserverns mjukvara efter eget behov. Eftersom den virtuella servern inte delas med någon annan finns det tillgång till mer kapacitet och risken för driftstörningar är mindre jämfört med en delad server såsom ett webbhotell. En VPS är dock

dyrare än ett webbhotell och kräver mer arbete av användaren eftersom denne konfigurerar webbservern.

En dedikerad server är liknande en VPS förutom att det inte finns någon virtuell server. Ägaren får givetvis möjlighet att konfigurera webbservern helt efter eget behov. Även här används servern enbart av ägaren, vilket leder till ett ökat pris, högre kapacitet och lägre risk för driftstörningar jämfört med användning av webbhotell.

2.3.2 Implementering av ett webbgränssnitt

Vid skapande av ett webbgränssnitt används flera olika tekniker och ramverk i kombination för att skapa ett visst utseende eller en viss funktionalitet. För uppritning av ett gränssnitt är det vanligt att använda ett märkspråk [20]. Det är ett språk som använder beskrivningstaggar för att hjälpa andra program presentera vad som ska visas, exempel på ett sådant språk är HTML, *HyperText Markup Language*.

Ett webbgränssnitt kan skapas på flera olika sätt. Vanligtvis brukar det finnas en databas för lagring av data, en backend för funktionalitet och en frontend för design av gränssnittet. Komplexiteten på webgränssnittets delar skiljer sig beroende på vad för typ av gränssnitt som ska implementeras. Det är även vanligt att använda olika ramverk för att underlätta utvecklingen av ett webbgränssnitt.

Ett alternativ för att underlätta skapandet av ett webbgränssnitt är att använda ett webbpubliceringssystem. Det är ett system som låter användaren skapa, redigera och organisera ett gränssnitt utan att använda programmering [21]. Istället innehåller systemet färdiga komponenter som används för att skapa ett webbgränssnitt. Det är möjligt att kombinera ett webbpubliceringssystem med egenutvecklade komponenter. Detta görs för att använda systemets fördelar kombinerat med specifik funktionalitet som inte annars erbjuds.

2.3.3 Relations-, NoSQL-, och hierarkiska databaser

En databas är en organiserad samling data som är lagrad på ett sådant sätt att information kan läggas till, ändras och hämtas [22]. För att skapa en databas krävs en datamodell som beskriver vad för typ av data som databasen innehåller och vad för information som associeras med denna data. Ett objekt beskrivs ofta med lagrad objektrelaterad data. Ett exempel är en lista på personer vars egenskaper innefattar personnummer, kön, ålder, adress och telefonnummer. Databasen

kan vara uppbyggd på flera olika sätt, det finns till exempel: relationsdatabaser, NoSQL-databaser och hierarkiska databaser.

I en relationsdatabas lagras data i tabeller där varje tabell ses som en relation. Varje relation innehåller en grupp egenskaper, så kallade attribut, där varje attribut är en egen kolumn i tabellen. Data hämtas ur tabellen med hjälp av relationsalgebra. Språket SQL, *Structured Query Language*, har utvecklats för att kunna interagera med databasen på ett användarvänligt sätt [23].

NoSQL är ett samlingsnamn för en typ av databas som sparar data ostrukturerat för att låta programmeraren, med hjälp av olika algoritmer, bestämma hur data kan hittas. NoSQL-databaser är flexibla och designas för att kunna använda olika typer av datastrukturer. De används främst vid lagring av stora mängder data då de har förmågan att breda ut sig över flera servrar samtidigt som de alltid är tillgängliga för läsning [24].

I en hierarkisk databas lagras information i en trädstruktur med länkar mellan noder innehållandes data. Varje nod har fält där data lagras, likt relationsdatabasens attribut. Mellan en föräldranod och dess barn kan det endast finnas en-till-en eller en-till-flera-relationer, vilket medför problem om ett attribut har flera föräldrar vilket då skulle representeras av en flera-till-en-relation [23].

2.3.4 MySQL och SQLite

Interaktionen med databasen sker med hjälp av ett generellt databashanteringssystem som är oberoende av vilken applikation som använder databasen. Exempel på sådana hanteringssystem är MySQL och SQLite.

MySQL är en databashanterare som ingår i programpaketet LAMP, *Linux Apache MySQL PERL/PHP/Python*, och används bland annat i webbapplikationer [25]. MySQL tillåter en hög grad av konfiguration. Bland annat går det att specificera olika användarkonton med specifika rättigheter att utföra operationer på databasen [26].

SQLite är en databashanterare som integreras och blir en del av applikationen som den implementeras i [27]. SQLite kan användas i olika applikationer så länge det är en ensam process som kommunicerar med databasen.

2.3.5 SSH: Secure Shell

SSH är ett protokoll som används för fjärranslutning till ett annat system över internet. SSH har stort fokus på säkerhet och all kommunikation mellan värd och klient krypteras. Detta betyder att det inte ställs några säkerhetskrav på nätverket.

Med SSH ges användare möjlighet att ansluta och logga in på en fjärrvärd och exekvera kommandon. Administratören behöver därmed inte vara på plats för att felsöka ett system och en användare kan få tillgång till sina filer från vilken plats som helst. Detta innebär givetvis också en säkerhetsrisk; om fel person får tillgång till värddatorn kan detta leda till allt från förlust av data till spridning av känslig information eller att systemet rent av blir obrukbart [28].

2.4 Bildbehandling

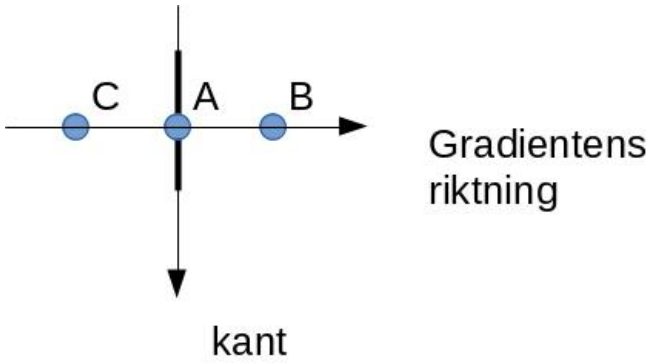
Med hjälp av avancerade matematiska algoritmer är det möjligt för ett datorprogram att med hjälp av datorseende-tekniker simulera det mänskliga ögat och hitta detaljer i en bild. Nämnvärda exempel på fält där datorseende-tekniker används är vid inspektion av flygplansdelar, automatisk översättning av handskrivna adresser på brev till digitaliserad text hos postföretag, samt fingeravtrycksläsare [29].

SimpleCV är ett ramverk som är byggt för att använda bibliotek som utnyttjar datorseende, till exempel OpenCV. SimpleCV är skrivet i programspråket Python och används genom att anropa ramverkets funktioner [30]. De förenklade funktionerna konverteras sedan i ramverket till mer avancerade funktioner som finns i något av de tillhörande biblioteken. Detta tillvägagångsätt är användbart i situationer där den analys som användaren vill åstadkomma inte är allt för komplex, då SimpleCV inte kräver djupare teoretisk kunskap i avancerad bildanalys som hade varit nödvändig vid användning av mer avancerade ramverk.

2.4.1 Canny kantdetektion

Canny kantdetektion är en serie operationer som utförs på en bild för att hitta skarpa kanter [31]. Det är en metod i fyra steg som till att börja med förutsätter att bilden som analyseras är i gråskala. Det första steget är att applicera ett Gaussiskt filter som sänker skärpan i bilden och på så sätt reducerar bruset i bilden. Nästa steg är att beräkna en storlek och en riktning för en gradient, se figur 2.2. Detta görs

för samtliga pixlar i bilden genom matrisberäkningar. Gradienten som beräknas kommer alltid vara vinkelrät mot kanten i bilden.



Figur 2.2: *Gradientens riktning i förhållande till en kant då en bild analyseras med Canny kantdetektion.*

Den tredje operationen är att kontrollera vilka punkter vars gradient utgör ett lokalt maximum. I figur 2.2 är punkt A på en kant och punkt C respektive B är närliggande punkter. Punkt A kommer i detta fall utgöra ett lokalt maximum och de andra två punkterna kommer att förkastas. De punkterna som är kvar i bilden nu skickas vidare till nästa operation som utför en hystereskontroll på punkterna som är kvar. Vid denna hystereskontroll specificeras två värden, ett maxvärde och ett minvärde. De punkterna som har en gradient större än maxvärdet anses med säkerhet vara kanter. De punkter med en gradient under minvärdet anses med säkerhet ej vara kanter. Punkterna som ligger inom max-min intervallet anses vara del av en kant om de är anslutna till en punkt som är över maxvärdet. Algoritmen resulterar i en binär bild vars kanter representeras av en etta (vitt) och det som inte är kanter är noll (svart) [31].

2.4.2 Blobdetektion

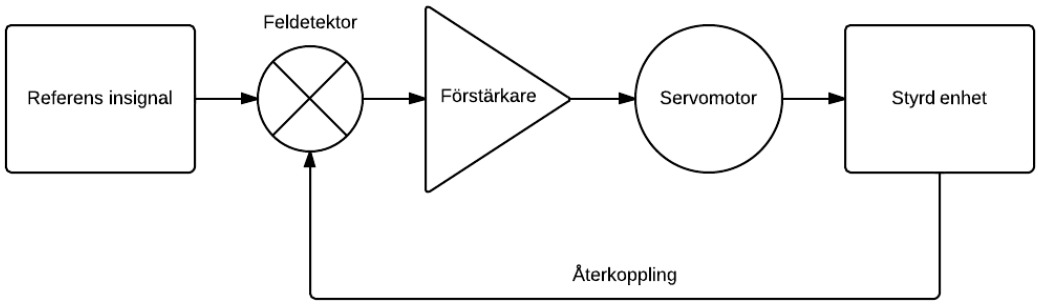
En blob är en term i datorseende-sammanhang som syftar på ett sammanhängande område i en digital bild. Området kan ha egenskaper som till exempel färg eller ljusstyrka. Genom att söka efter blobs i en bild är det möjligt för en dator att autonomt välja ut intressanta områden ur en bild och därefter använda informationen som identifieras i dessa områden [32].

En metod för att identifiera blobs är att göra om en bild till gråskala och analysera bildens pixelvärden. Dessa värden sammanställs och pixlarna försöker delas in i ett antal grupper baserat på deras värden. Dessa grupper analyseras och genom

att försöka minimera pixelvärdenas varians inom gruppen, och därmed maximera variansen mellan grupperna, kommer de optimala tröskelvärdena att kunna beräknas. Denna metod för att beräkna tröskelvärden benämns Otsus metod [33]. När pixlarna delats in i grupper kontrolleras vilka pixlar som ingår i samma grupp och ligger intill varandra. Dessa pixlar kategoriseras i en ny grupp och bildar en blob.

2.5 Servomotor

En servomotor består huvudsakligen av tre komponenter: en elmotor, en kontrollkrets och en lägessensor för motorn. I enklare servomotorer är denna lägessensor en enkel potentiometer som är ansluten till elmotorns axel. Ett servo kan modelleras som ett första ordningen återkopplat reglersystem, se figur 2.3. En referensposition skickas in i systemet som jämförs med den aktuella positionen och felets storlek används som insignal till motorn och styr den mot referenspositionen [34].



Figur 2.3: *Förenklad representation för servot i form av ett blockschema, för ett första ordningens återkopplat reglersystem.*

Styrsignalen till servot är en fyrkantsvåg med en frekvens på 50 Hz. En duty-cycle på 10 % ger ett utslag på +90 grader, och 5 % ger ett utslag på -90 grader. Detta innebär att en signal ska genereras med en positiv flank varje 20 ms. Om vi söker ett utslag på +90 grader ska den negativa flanken komma 2 ms efter den positiva flanken [35].

3

Systemdesign

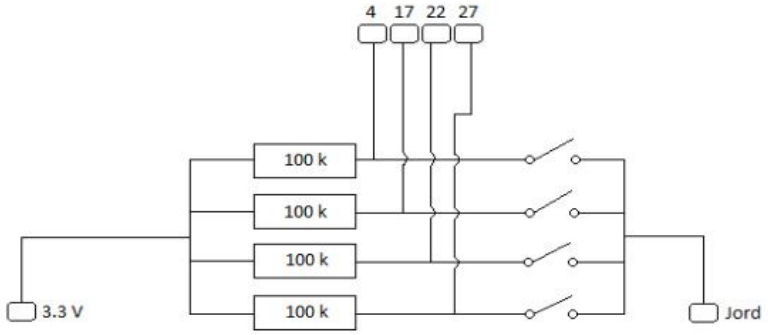
I detta kapitel förklaras vilka designval som genomförts vid implementering av E-blackboard.

3.1 Installation av sensorer

Mekaniska brytare har valts för att meddela Raspberry Pi:n när en tavla är färdigskriven. Dessa valdes till följd av att de är lätthanterliga och tillgängliga utan kostnad. Projektets sensorer liknar de mekaniska brytare som används i dagens mekaniska tangentbord [36].

För att konstruera en krets som kan hantera de två tavlorna per tavelkolumn, se figur 1.1, behövdes en brytare per tavla. Brytarna monterades högst upp på tavelkolumnen och sladdarna drogs längs med taket i en kabelränna till Raspberry Pi:n.

Kretsen med brytarna matades med spänning från ett av 3,3 V-stiften på Raspberry Pi:n, se figur 3.1. Till denna källa parallellkopplades sedan fyra motstånd på 100 kΩ, följt av en brytare vardera och sedan kopplades allt till jord. Innan varje brytare kopplades en sladd till ett insignal-stift. Med denna konfiguration kommer insignalen för respektive brytare till Raspberry Pi:n vara låg då brytaren sluts och hög då den är öppen.



Figur 3.1: *Kretsen med markerade kopplingar på signalfördelningen till var och en av de fyra detektion pinnarna på Raspberry Pi:n och till respektive brytare. De fyra brytarna är kopplade till samma jord på Raspberry Pi:n.*

3.2 Webbplats

Nedan presenteras de val som gjordes vid uppbyggnad av webbplatsen. Detta inkluderar val av webbserver, utveckling av webbgränssnittet, konstruktion av databas samt upprättande av SSH-tunnel.

3.2.1 Webbserver

Webbhotellet Binero valdes för publicering av webbgränssnittet då funktionaliteten och kapaciteten som erbjuds väl täcker in de behov som identifierats under projektets gång. Några relevanta funktioner som erbjuds i privatpaketet är 100 GB lagringsutrymme, 2000 GB trafik/månad samt SSH-möjlighet och lätthanterliga verktyg för webbgränssnittet [37].

3.2.2 Webbgränssnitt: Utveckling och design

Vid utvecklingen av webbgränssnittet var tanken att göra det så enkelt som möjligt. Det ska vara lätt att hitta föreläsningsanteckningar för en bestämd dag och kurs redan första gången gränssnittet besöks. Framförallt ska det vara så få knapptryck som möjligt för att få fram de sökta bilderna. För att uppnå detta krävdes inga avancerade funktioner utan enbart koppling mellan frontenden och databasen

och därmed implementerades ingen backend. PHP användes för kopplingen mellan frontenden och databasen då detta är en väl beprövad teknik med utmärkt dokumentation på internet.

För design av webbgränssnittet har tre ramverk använts: Wordpress, Bootstrap och Zoomer. Wordpress är ett webbpubliceringssystem med stor användarbas [38]. Systemet tillåter skapandet av ett eget tema som sedan appliceras på samtliga sidor som skapas. Detta underlättar utvecklingen av webbplatsen då komponenter som ska se ut på samma sätt över hela webbplatsen samlas på ett ställe. Dessa egenskaper i kombination med ett intresse att utforska möjligheterna med Wordpress låg till grund för valet att använda detta system. Bootstrap är ett ramverk där färdiga komponenter såsom knappar och tabeller skapats med språken CSS och Javascript [39]. Tidigare användning av Bootstrap har visat att det sparar mycket tid och producerar ett gott resultat vilket låg till grund för valet att använda ramverket. Det sista ramverket, Zoomer [40], används för att möjliggöra zoomning på bilderna som presenteras på gränssnittet. Det finns många olika ramverk ute på internet med zoomfunktioner och Zoomer valdes tack vare att den hade de egenskaper som söktes till gränssnittet.

Vidare följer webbgränssnittet tre designmönster för att underlätta navigering så att användaren enkelt kan hitta en specifik anteckning. Designmönstret Center Stage har använts, som är anpassat för ett gränssnitt vars primära uppgift är att upplysa om en specifik funktion [41]. Enligt designmönstret ska funktionen som ska lyftas fram täcka största delen av gränssnittets huvudsida. Gränssnittets första sida består till största delan av en lista med alla kurser som gör det lätt för användaren att hitta de föreläsningsanteckningar som efterfrågas. För presentation av bilderna användes One-Window Drilldown som är rekommenderat att använda då utrymmet är begränsat eller om den intressanta informationen är så stor att den behöver all plats den kan få. Enligt designmönstret ska objekten visas i en vertikal lista och fylla ut hela sidan. Eftersom bilderna kan innehålla små figurer och texter behövde de täcka större delen av sidan för att användaren ska kunna se vad som står på tavlorna. Det sista designmönstret som användes var Escape Hatch, som bör användas när användaren behöver gå igenom en process för att nå sitt mål. Då användaren både behöver välja kurs och därefter datum implementerades designmönstret för att användaren enkelt ska kunna börja om från början.

3.2.3 Databas

Det finns flera olika databaser att välja mellan men till följd av tidigare erfarenheter inom relationsdatabaser minskades antalet alternativ avsevärt. Till en början valdes SQLite som databas då det är en databashanterare som är enkel att ad-

ministrera och är resurssnål. Dock förkastades idén att använda SQLite eftersom den inte lämpas till större applikationer. MySQL valdes istället då webbhotellet erbjöd stöd för denna databas och tillhandahöll även verktyget phpmyadmin som kan användas för att administrera MySQL-databaser.

Till en början bestod databasen enbart av en tabell med fyra kolumner som var associerade med en specifik bild som laddades upp till webserven från Raspberry Pi:n. Varje rad innehöll följande information: ett unikt ID för bilden, en sökväg för bilden, ett datum när bilden togs samt kurskoden för den kurs som bilden tillhör.

För att söka upp den information databasen behöver hämtas en prenumerationsfil i iCalendar-format från schemasystemet TimeEdit. Vid projektets start fanns inte all den sökta informationen tillgänglig, däremot hittades ett unikt identifikationsnummer som gjorde det möjligt att extrahera resterande information från TimeEdits webbgränssnitt.

Under projektets gång skedde en förändring i prenumerationsfilens innehåll och i dagsläget kan samtlig information extraheras från denna fil. Ett nytt attribut samt en ny tabell infördes för att möjliggöra identifiering av bilder med bristfällig kvalité samt för att lagra kursnamn.

3.2.4 SSH: Secure Shell

Då det beslutades att låta webbhotellet vara värd för databasen uppmärksammades ett nytt problem; alla anslutningar till databasen var tvungna att komma inifrån webbhotellet. Detta innebar att vår Raspberry Pi inte kunde ansluta direkt till databasen. För att kringgå problemet konfigurerades en SSH-tunnel som gjorde att data som skickades på en specifik port på Raspberry Pi:n gick genom en SSH-server på webbhotellet och vidare till databasen.

3.3 Bildbehandling

Det största problemet med att klippa ut en tavla från en bild är att få programmet att autonomt avgöra vilka delar som tillhör tavlan och vilka delar som är irrelevanta i bilden. Eftersom ett av projektets avgränsningar var att inte skriva helt egna funktioner för bildbehandlingen gjordes en kortare förundersökning bland olika ramverk som skulle kunna användas. SimpleCV valdes ut tidigt, då det visade sig att detta ramverk är ett kraftfullt verktyg med många nyttiga funktioner som skulle komma väl till hands samtidigt som det är relativt lätt att arbeta med.

3.3.1 Metod 1: Med markörer

Till en början användes funktionen *FindBlobs* för att hitta sammanhängande områden i bilden, se kapitel 2.4.2 för vidare förklaring om hur detta går till. Eftersom att en bild kan skilja mycket i till exempel ljusförhållanden beroende på när bilden tas, kan närliggande pixlar skilja sig till den grad att funktionen inte ser dem som sammanhängande trots att det mänskliga ögat gör det.

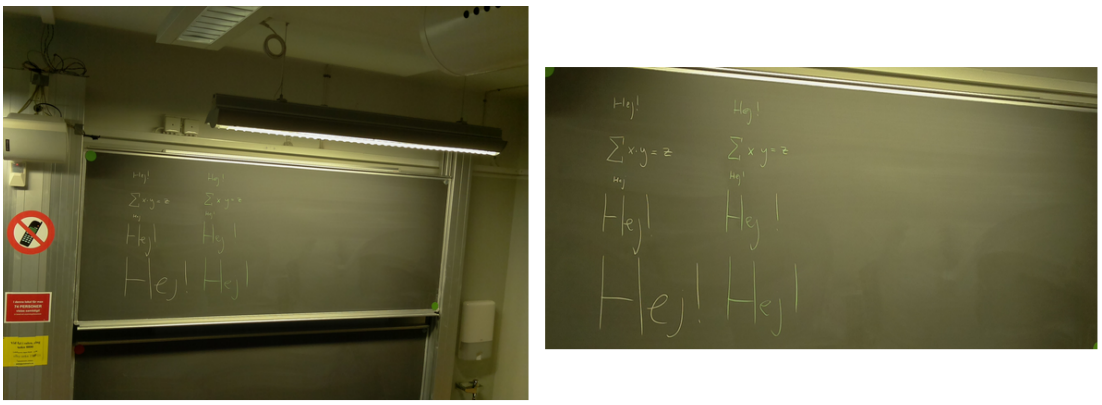
Genom att segmentera färgerna är det möjligt att räkna ut avståndet mellan en önskad färg och samtliga färger på en bild som mäts i RGB-skalan. SimpleCV har en funktion, *ColorDistance* [42], som utnyttjar detta. Den önskade färgen skickas in i programmet där uträkningar sker som jämför denna färg med samtliga pixlar på bilden. En ny bild i gråskala returneras där de färger som ligger nära den önskade färgen färgas mörkt, samtidigt som de färger som ligger långt ifrån den önskade färgen returneras som vitt. Figur 3.2 visar hur en föreläsningstavla med gröna markörer har körts igenom funktionen *ColorDistance* med grön färg som inparameter. Markörerna blir betydligt mörkare än resten av bilden, då dess färger stämmer väl överens med inparametern.

Figur 3.2: *En bild på en tavla efter funktionen ColorDistance som svartmålat de gröna markörerna.*

För att ytterligare ta bort de delar i bilden som inte är av intresse men som ligger nära den önskade färgen användes funktionen *binarize*, som gör om alla färger på den gråskalade bilden till antingen svart eller vitt. Detta görs med hjälp av en tröskelvariabel som sätts som gräns, där de färger som ligger över variabeln blir svartfärgade och resten blir vita [43]. För att detektera tavlan användes markörer av olika färger i två av tavlans hörn för att lättare upptäcka tavlans position. Programmet letade efter markörernas färger med hjälp av tidigare nämnda funktioner

och sparade pixelvärdena i variabler som sedan användes för att klippa ur bilden mellan pixlelvärdena.

Efter test 2, se bilaga B.2, av systemet visade sig lösningen inte vara optimal eftersom ljusförhållandena i salen skiftar mycket beroende på yttre omständigheter. Resultaten blev därmed högst inkonsekventa och en ny lösning började utformas som inte var beroende av markörer eller specifika ljusförhållanden. Ett annat problem som uppmärksammades vid testet var att bildens perspektiv gjorde tavlan mer svårläst än vad som önskas, se figur 3.3. Vänster del av figuren visar den oredigerade bild som togs av kameran, höger del är den urklippta tavlan. Funktionalitet för att korrigera perspektivet började utvecklas efter test 2.



Figur 3.3: *Resultatet från den första versionen av bildbehandlingen med utplacerade markörer.*

3.3.2 Metod 2: Utan markörer

En ny strategi utformades för att angripa problemet. Processen finns illustrerad steg för steg i bilaga C. Det första steget i processen är att använda funktionen *Edges* som innebär att Canny kantdetektion används, en metod som beskrivs i kapitel 2.4.1. Det som returnerars är en svartvit bild där identifierade kanter är vita och allt annat är svart. Förutsatt att kameran är centrerad på tavlan kommer det finnas ett område i mitten av bilden som går att separera från resten av bilden. För att säkerställa att området är ordentligt avgränsat används en funktion som förstorar vita pixlar. Separationen sker sedan genom att använda funktionen *floodFill* som specificerar en eller flera pixlar och färgar samtliga pixlar i samma område med en specifik färg [44]. Genom att specificera ett flertal pixlar inom tavelramen minimeras risken av att enbart träffa text eller figurer på tavlan, vilket skulle leda till att hela operationen misslyckas.

Bilden kommer efter denna process bestå av ett stort enfärgat område i mitten av bilden och en mängd tunna linjer som inte är av intresse. För att bli av med detta brus i bilden användes en funktion som benämns *Erode* [43]. Funktionen gör att mörka områden tränger in på ljusare områden, vilket resulterar i att alla linjer utanför bilden tas bort och området i mitten, som har samma form som tavlan i ursprungsbilden, finns kvar. Nästa steg är att identifiera det sammanhängande området i bilden. Dock är området inte tillräckligt väldefinierat för att fortsätta arbeta med. För att lösa detta problem används en funktion för att plocka fram områdets konvexa form, vilket resulterar i att områdets yttre kanter blir, åtminstone delvis, raka. Därefter används en funktion för att identifiera linjer i bilden, vilket innebär att använda en Hough-transform [45]. Genom att undersöka vilka linjer som har en mellanliggande vinkel inom ett specificerat intervall och var linjerna möts skapas fyra kluster av punkter som ligger i det område tavlans verkliga hörn befinner sig i. En algoritm som benämns k-means [46] separerar de fyra klusterna och beräknar deras mittpunkter vilket resulterar i koordinater som befinner sig i närheten av de verkliga hörnen. När vi slutligen har fått tag i denna information används en funktion för att beräkna en matris som används för att translatera hela bilden och på så sätt ändrar perspektivet. Translationen är en matrisoperation som innebär att varje bildpunkt i bilden förflyttas enligt ett specifikt mönster. Slutligen används de nya koordinaterna för hörnen för att klippa ut tavlan ut bilden och på så sätt exkludera all onödig information i bilden.

3.3.3 Tavelidentifikation

Sensorernas enda uppgift är att tala om för systemet när en tavla har nått sin översta position vilket leder till att systemets mjukvara måste avgöra om bilden som tagits innehåller den önskade informationen eller inte.

Systemet är programmerat att efter utslag från en sensor ta en bild och försöka köra den genom bildprocesseringen. Under processeringen görs ett antal tester för att säkerställa att resultatet som produceras är av hög kvalité. Det undersöks om den identifierade tavlan är tillräckligt stor, att fyra hörn identifierats, och om de identifierade hörnen är på ett rimligt avstånd från varandra. Om ett av testen skulle ge ett negativt resultat kommer en ny bild att tas och den nya bilden körs genom processen. Om det efter tre försök inte går att producera något godkänt resultat kommer programmet att avbryta processen och spara bilden för senare inspektion av en administratör.

3.3.4 Förhöjning av bildkvalitet med filter

För att en bild på en tavla ska vara behaglig att läsa är det viktigt att texten i bilden är tillräckligt skarp och att ljusförhållanden inte påverkar på ett negativt sätt. Vid test 2, se bilaga B.2, blev det uppenbart att bildkvaliteten var beroende av ljuskällor i rummet där testet utfördes, då varje bild kunde skilja sig mycket på grund av reflektioner och ljus i bakgrunden.

Genom att applicera ett filter som förstärker önskade delar av en fotograferad tavla går det att öka läsbarheten samtidigt som övriga delar av bilden inte förändras [47]. Det valda filtret, *edge_enhance*, använder sig av kantdetektion för att förstärka den skrivna texten på en tavla så att den sticker ut från sin omgivning mer och inte längre är lika känslig för skiftande ljusförhållanden.

För att normalisera den skiftande ljusstyrkan skapades en funktion som först valde ut en pixel från en av tavlans hörnmarkörer och tog ut ljusstyrkan genom att mäta genomsnittet av färgerna på pixeln delat med de tre färgkanalerna: röd, grön och blå. Med hjälp av detta värde bestäms sedan om kontrasten i bilden behöver sänkas eller höjas. Höjningar och sänkningar sköttes med hjälp av funktionen ImageEnhance, som med hjälp av enkla inparametrar modifierar bilder som skickas med i funktionen [48]. På grund av att bildanalysmetoden byttes ut försvann markörerna och en ny lösning för att implementera filtret hann inte göras.

3.4 Installation av servomotorn

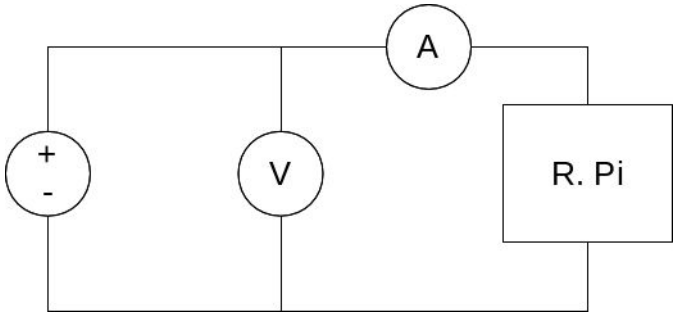
För att kunna ackommodera fler än en tavla i föreläsningssalen implementerades en servomotor för att kunna rikta om kameran. Flera alternativ undersöktes men snart stod det klart att servot var den smidigaste lösningen. Användningen av servomotorer i kombination med Raspberry Pi är relativt utbredd och det finns gott om elektronisk dokumentation tillgänglig för både enklare och mer avancerade lösningar. Kapitel 2.5 beskriver mer detaljerat hur en servomotor fungerar.

Styrsignalen till servomotorn genererades genom att Python funktioner styr GPIO-stift 18 på Raspberry Pi:n, se figur 2.1. Efter noga testning visade sig denna lösning inte helt optimal då Raspberry Pi:n inte med hjälp av mjukvara kan producera en tillräckligt exakt signal. Detta är ett problem som går att ta sig runt genom att flytta styrningen av servot närmre hårdvaran. På grund av viss tidsbrist har denna funktionalitet inte implementerats.

Då servot tidvis drar mer ström än vad Raspberry Pi:n kan producera [13], seriekopplades fyra AA batterier på Raspberry Pi:ns takfäste.

3.5 Systemets energiförbrukning

Systemet består av flera komponenter som ska samarbeta effektivt. En undersökning av systemets energiförbrukning utfördes för att ta reda på driftkostnaderna samt eventuella behov av optimeringar. Mätpunkterna som användes var på de ledare som ansluter Raspberry Pi:n till dess strömadapter, se figur 3.4. Till strömmätningen användes CHY 14 digitala multimeter och till spänningsmätningen Caltec Instrument CM 1100. Strömriktig koppling valdes då Raspberry Pi:n är högresistiv relativt utrustningen som mäter strömmen. Multimeterns inre resistans eftersträvar att vara försumbar relativt mätobjektet.



Figur 3.4: *Strömriktig koppling av mätutrustningen vid effektmätningen. Batterisymbolen motsvarar adaptern som kan avge 1 Ampere och 5 Volt. Symbol A representerar CHY 14 digital multimeter och symbol V representerar Caltec Instrument CM 1100 digital multimeter.*

För att få en uppfattning om hur mycket energi som förbrukades vid systemets användning noterades mätvärden var 5:e sekund då följande aktiviteter skedde: uppstart, aktivering av en sensor, vänteläge och avstängning. Under hela strömmätningen var spänningen mestadels stabil på 5 V och strömmen varierade mellan 0 A och 0,65 A. I figur 3.5 presenteras mätdata från strömmätningen och på tabellform i bilaga B. Tabell 3.1 visar de aktiviteter som utfördes under mätningen. Aktiviteterna finns utmärkta i figur 3.5 som vertikala markeringar.

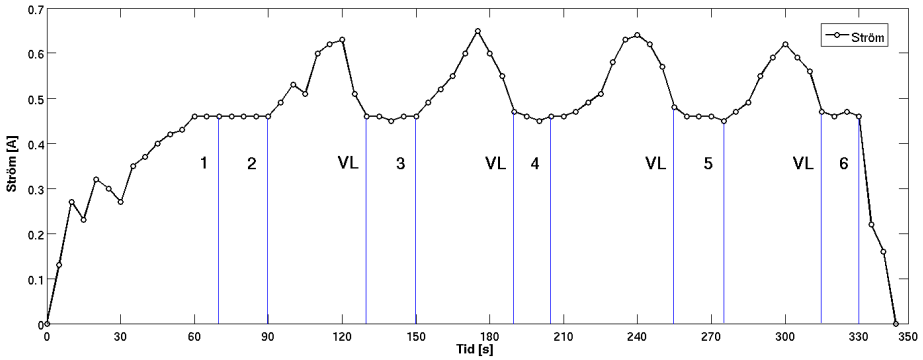
Den förbrukade energin beräknades enligt ekvation 3.1, där U är spänningen, I är strömmen och t är tiden i timmar. Resultatet blir den förbrukade energin i mWh.

$$E = (U * I * t) * 1000 \tag{3.1}$$

För att beräkna energiförbrukningen vid de olika aktiviteterna beräknades funktionens area genom att integrera över funktionen med hjälp av trapetsmetoden.

Tabell 3.1: *De olika aktiviteterna under strömmätningen*

Aktivitet
(1) Uppstart slutförd
(2) Sensor 1 aktiverad
(3) Sensor 2 aktiverad
(4) Sensor 3 aktiverad
(5) Sensor 4 aktiverad
(6) Avstängning påbörjad
(VL) Vänteläge aktiverad



Figur 3.5: *Mätdata från strömmätningen på Rasperry Pi:n med vertikala markeringar över de olika aktiviteterna.*

Genom att approximera en linjär förändring mellan två punkter och sedan integrera över denna linjärisering beräknas arean under kurvan. Arean under kurvan är den energi som förbrukas under dess tidsintervall. Aktivitetens energiförbrukning är summan av de areor vars regioner finns inom aktivitetens livstid [49].

4

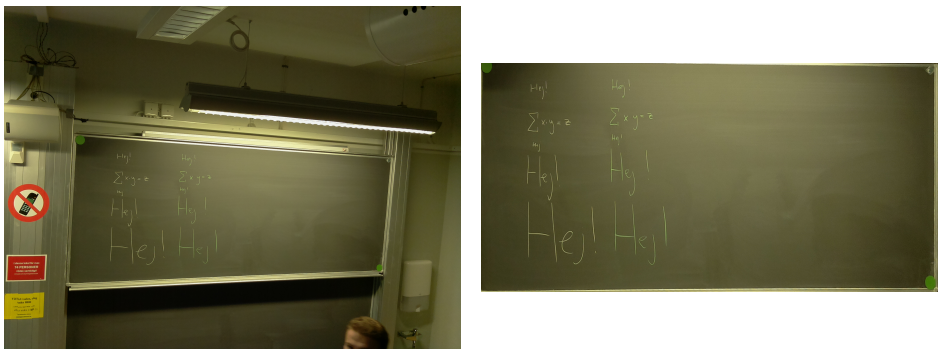
Resultat

Tillsammans bildar delarna sensor, webbplats, bildbehandling, databas och motor ett system som kan digitalisera föreläsningsanteckningar. Vid projektstart utformades fyra delproblem: övergång från den analoga krittavlan till ett digitalt format, kvalitetssäkring av bilden, publikation av informationen på ett webbgränssnitt samt att systemet inte påverkar eller stör under en föreläsning.

4.1 Prototyp

Det första problemets målsättning, att all information som föreläsaren skrivit ner ska finnas digitaliserat, är inte fullt uppnått. Det uppstår problem om något objekt befinner sig mellan kameran och tavlan när en bild tas. Om detta inträffar avbryter bildanalysen processen och systemet tar en ny bild och försöker köra bildanalysen med den nya bilden. Om processen misslyckas tre gånger i rad, publiceras inte tavlan. Åtgärder har dock vidtagits för att undvika att detta händer. Kameramodulen är placerad i taket och bilder tas enbart när tavlan är i dess högsta position, vilket minskar risken att något objekt befinner sig mellan kameran och tavlan när en bild tas. En servomotor, som låter kameramodulen rotera runt sin fästpunkt i taket, har implementerats. Rotationen gör att systemet kan digitalisera all information som presenteras under en föreläsning, oavsett vilken tavla föreläsaren väljer att använda. Till följd av att servomotorn implementerades så uppstod ett annat problem: alla bilder tas från en vinkel. Detta problem ledde till utvecklingen av en funktion

för automatisk redigering av bilderna. Den automatiska redigering kombinerades med en analys som kan urskilja tavlan från bakgrunden och göra den korrigering av perspektivet som krävs i samband med att tavlan klipps ut från bakgrunden. I figur 4.1 visas en bild innan och efter den gått genom den automatiska bildredigeringen. Det implementerades även en kontroll som tillåter cirka 30 % av tavlans originalstorlek vara dold av objekt, vilket innebär att pekpinnar eller andra liknande hjälpmedel som föreläsare använder inte påverkar bildanalysen. Då det fanns begränsad prestanda att arbeta med var det viktigt att programmets exekvering inte tog för lång tid. I tabell 4.1 finns resultaten från en undersökning av hur lång tid exekveringen av bildbehandlingen tog för bilden som presenteras i figur 4.1. Resultatet indikerar att bildbehandlingen inte är för resurskrävande för att köras på Raspberry Pi:n.



Figur 4.1: *En bild före och efter automatisk redigering.*

Det andra problemets målsättning, att bilder som presenteras på webbgränssnittet enbart visar upp tavlan, är inte heller fullt uppnått. Samma problem som ovan kan påverka resultatet. Om en föreläsare är över två meter lång och står precis intill tavlan kan delar av huvudet finnas med i bilden som visas på gränssnittet. Situationen kan anses vara ett undantagsfall och kan omöjligt inträffa i någon av skolans större föreläsningssalar.

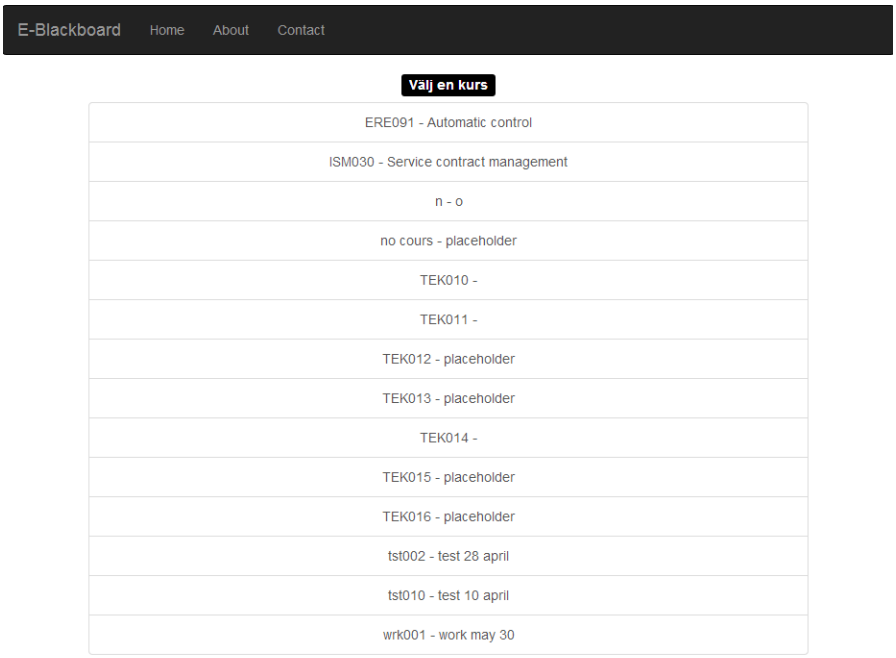
Det tredje problemets målsättning, att implementera ett webbgränssnitt som är anpassat för att hjälpa användaren hitta föreläsningsanteckningar anses vara uppnått. Vid implementering av gränssnittet har olika designmönster använts som stödjer målsättningen. Bland annat har designmönstrena Center Stage och One-Window Drilldown använts, som beskrivs i kapitel 3.2.2. Genom att användaren möts av en lista med alla kurser blir det tydligt hur användaren ska hitta föreläsningsanteckningarna. Figur 4.2 visar hur gränssnittets framsida ser ut.

Tabell 4.1: *Undersökning av exekveringstid för den automatiska bildbehandlingen med en exempelbild på Raspberry Pi.*

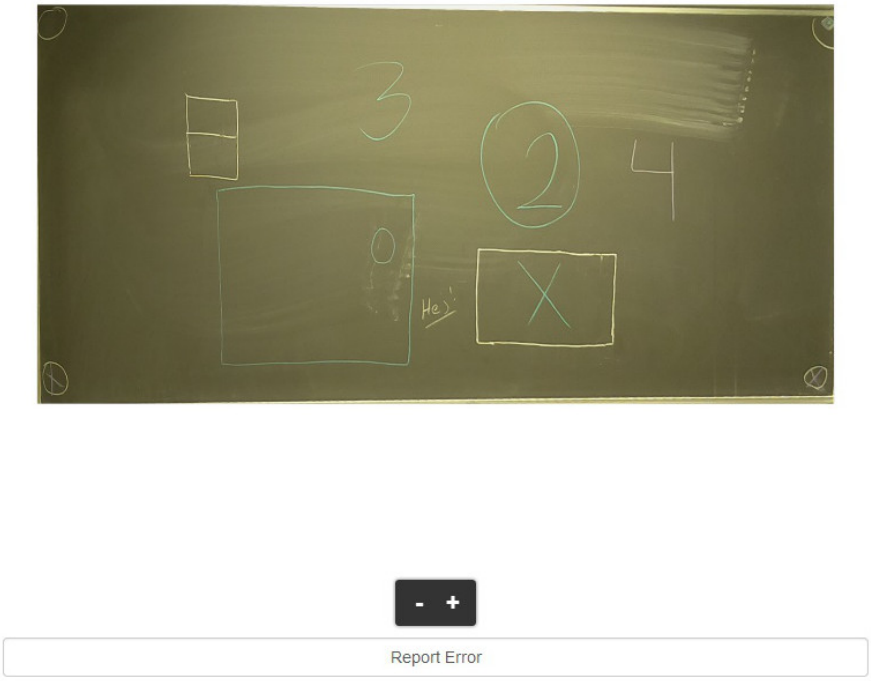
Funktion	**Tid [s]**
Öppna bild tagen av kameran	2,28
Kantdetektion	1,88
Dilate - förstorning av vita bildpunkter	1,24
FloodFill - utfyllnad av tavlan (9 sampel)	1,99
Erode - ta bort tunna linjer	3,87
FindBlobs - identifiera sammanhängande områden	2,04
Ta fram områdets konvexa form	0,27
Identifiera linjer	2,20
K-means - identifiera och beräkna centrum för kluster	0,43
Korrigera perspektivet	9,32
Spara resultatbild	0,97
Övriga funktioner	0,37
Total tid	**26,86**

One-Window Drilldown användes eftersom bilderna som presenteras på webbgränssnittet behöver täcka större delen av sidan för att användaren ska kunna se vad som står på tavlorna. Figur 4.3 visar hur en bild ser ut när den presenteras på webbgränssnittet.

De två sista målsättningarna, att processen från bildtagning till publicering på webbgränssnittet aldrig ska ta längre tid än tre minuter samt att systemet ska fungera autonomt efter att en sensor blivit aktiverad, har båda uppnåtts. När bildbehandlingen lyckas på första försöket tar en körning ungefär 40 sekunder. Om problem påträffas under bildbehandlingen kan det hända att processen behöver köras tre gånger. I detta fall kommer hela processen ta ungefär två minuter, vilket lämnar god marginal till gränsen på tre minuter.



Figur 4.2: *Startsidan för webbplatsen.*



Figur 4.3: *Bildpresentation på webbplatsen.*

4.2 Energiförbrukning

Beräkningar på energiförbrukningen, för vardera av de olika aktiviteterna, sammanställs i tabell 4.2 nedan.

Tabell 4.2: *Energiförbrukning för respektive aktivitet.*

Aktivitet	**Energiförbrukning [mWh]**
(1) Uppstart slutförd	32,2
(2) Sensor 1 aktiverad	30,2
(3) Sensor 2 aktiverad	30,7
(4) Sensor 3 aktiverad	37,8
(5) Sensor 4 aktiverad	30,0
(6) Avstängning påbörjad	4,24
(V) Vänteläge aktiverad	64,9
Medelvärde av sensor 1-4	32,2

Beräkningar från tabell 4.2 används sedan för att uppskatta energiförbrukningen under en hel skoldag, som presenteras i tabell 4.3. Antagandena att en föreläsning tar två timmar, att en sensor aktiveras 15 gånger per föreläsning och att det under en dag hålls fyra föreläsningar har gjorts.

Tabell 4.3: *Daglig energiförbrukning. En dag innebär fyra lektioner där 15 bilder tas per lektion.*

Aktivitet	**Tid [HH:MM:SS]**	**Energiförbrukning [mWh]**
Aktiverade sensorer	00:40:00	$1{,}931 * 10^3$
Stand by	08:48:35	$20 * 10^3$
Uppstart	00:01:10	32,20
Avstängning	00:00:15	4,240
Off	14:30:00	0
Summa	24:00:00	$22{,}27 * 10^3$

Energiförbrukningen för en skoldag används sedan för att beräkna den uppskattade kostnaden för ett system i en föreläsningssal under ett år. Under lov, helger och

omtentaperioder antas det att systemet är avstängt. Ett exempelpris på elkostnad tas från Göteborgs Energi DinEl (2014-05-05), se beräkning 4.1. Exempelpriset är beskrivet i kWh vilket medför att summan av energiförbrukningen måste görs om till kWh.

$$0.02227\ kWh * 171\ dagar * 0.8038\ kr/kWh = 3{,}06\ kr \tag{4.1}$$

Vid beräkning 4.1 har uppskattningar varit i överkant av verkliga värden och vi kan därmed med god säkerhet konstatera att systemet i drift kommer att kosta mindre än vad som redovisats här.

5

Diskussion

I detta kapitel diskuteras de förbättringar som bör implementeras för att realisera prototypen i verkligheten. Vidare utförs en jämförelse med liknande tekniker som används till samma ändamål. Slutligen diskuteras huruvida E-blackboard underlättar studenters inlärning eller inte.

5.1 Motiveringar baserat på resultat

Två av projektets delmål som nämns i kapitel 1.4 har inte uppnåtts helt. E-blackboard kan inte garantera att alla anteckningar alltid sparas, samt att en publicerad bild enbart innehåller krittavlan. För att uppnå målet att enbart få med tavlan på en publicerad bild med det nuvarande systemet skulle tröskeln för att systemet ska godkänna bilden, som är 70 % av tavlans storlek, kunna ökas. Detta hade dock lett till att bilder som enbart blockeras av något mindre störningsmoment eventuellt hade stoppats av storlekskravet.

Genom att öka antalet nya bildtagningsförsök, hade risken att tavlor går förlorade minskats. Däremot skulle en ökning av antalet bildtagningsförsök kunna medföra att tidsmålet inte hade nåtts. En ny föreläsningstavla skulle hinna bli färdigskriven och dess sensoraktivering hade inte registrerats av systemet överhuvudtaget, vilket inte går bra ihop med målet att alla anteckningar ska sparas.

Vi insåg snart att dessa två mål kunde säga emot varandra och bestämde oss för

en kompromiss där inget av dessa två mål nås fullständigt. Om flera algoritmer för att hitta objekt undersökts är det möjligt att vi hade hittat en lösning för att till exempel enbart blockera bilder där huvuden finns i bilden med hjälp av igenkänning av specifika former. Tidskravet hade möjligen även kunnat optimerats eller tillochmed eliminerats om någon form av kösystem hade implementerats som köar en bildtagning av en tavla vars sensor aktiverats under pågående bildredigering. Dessa lösningsförslag undersöktes aldrig på grund av projektets tidsbrist.

5.2 Från prototyp till produkt

Projektet har resulterat i en prototyp som nästan når upp till de krav man skulle kunna ställa på ett system som ska installeras i samtliga föreläsningssalar på Chalmers. Det krav som är viktigast att nå, och som inte fullt ut kan garanteras att det uppnås i dagsläget, är att all information som presenteras på tavlorna under en föreläsning faktiskt kommer upp på webbplatsen. Den begränsade garantin beror till stor del på att resultatet är beroende av föreläsarens interaktion med systemet. För att lösa detta problem är det viktigt att alla föreläsare blir introducerade till hur systemet arbetar och därmed kan skapa sig en förståelse för hur det ska användas. Om föreläsaren gör precis så som vi tänkt kommer vi kunna garantera att systemet, under normala omständigheter, alltid fungerar.

Om systemet ska implementeras i en större skala bör ett antal mindre förändringar på systemet genomföras. Det är viktigt att systemet inte stör föreläsningen, och därför bör servomotorn bytas ut mot ett alternativ som är tystare än det som används i dagsläget. Utöver servot måste en mer hållbar lösning implementeras för strömförsörjningen. Det hade varit fördelaktigt att driva både servo och Raspberry Pi från samma strömkälla för att säkerställa kontinuerlig drift. Det bör också ses till att Raspberry Pi:n som placeras ut i salarna är fastlåst då den kan anses stöldbegärlig. Slutligen bör det undersökas hur systemet bör installeras i de salar där tavlorna inte är rörliga. I dessa salar hade det till exempel kunnat installeras en knapp som föreläsaren trycker på för att aktivera bildtagningen. Förslaget är dock något som kan anses suboptimalt då det innebär att ett nytt moment behöver introduceras till en föreläsares rutin. En dedikerad server bör troligtvis användas om systemet börjar användas i större utsträckning men det bestämdes att lägga tid på andra delar av projektet istället för att upprätta och underhålla en server.

Vidare hade det varit fördelaktigt att integrera presentationen av informationen med de etablerade lärplattformar som används på Chalmers. Plattformen Ping-Pong har börjat användas mer och mer på senare tid för att publicera diverse information relaterade till specifika kurser. Genom att integrera vårt system med

PingPong kan all information som relateras till samma kurs centraliseras, något som uppskattas av slutanvändaren. För att förbättra användarupplevelsen kan ytterligare funktionalitet läggas till på webbgränssnittet som till exempel möjlighet att kommentera bilder. Bildkommentarer skulle kunna användas för att göra anmärkningar på eventuella felaktigheter i bilderna eller för att förklara innehållet då eventuella oklarheter identifieras. Funktionen har dock inte implementerats eftersom det lätt kan leda till missbruk. Det bästa hade varit om föreläsaren varit involverad i gränssnittet och haft ensamrätt att utveckla och förklara bilder som är svåra att förstå.

En ytterligare möjlighet att skapa värde för slutanvändaren är att utveckla en mobilapplikation som kan presentera bilderna från webbgränssnittet. I dagsläget är det många studenter som använder surfplattor och mobiltelefoner för att läsa dokument när de studerar. Att ha en applikation för att snabbt kunna navigera till information som presenterats under föreläsningar skulle vara värdefullt för många studenter.

Prototypens låga energiförbrukning resulterar i en låg årlig driftkostnad, vilket medför att inga vidare optimeringar på energiförbrukningen krävs för en slutgiltlig produkt.

5.3 Alternativa tekniker

För att E-blackboard ska vara en attraktiv lösning för skolor måste systemet kunna mäta sig med, eller vara kompatibelt med, existerande system. Det valdes att försöka hålla kostnaderna för systemet så låga som möjligt utan att förlora funktionalitet. Raspberry Pi är en billig enkortsdator med relativt låg prestanda som följd, men den erbjuder tillräcklig prestanda för vår applikation. De sensorer som används är väldigt enkla mekaniska brytare, men de uppfyller de krav vi ställer på våra sensorer och därmed finns det ingen anledning att lägga pengar på dyrare komponenter. Minneskortet som agerar lagringsenhet för Raspberry Pi:n är 16 GB stort och i dagsläget är det mer lagring än vad som behövs. Det är möjligt att denna lagringsvolym kan komma till användning om mer avancerad funktionalitet ska läggas till, så som automatisk jämförelse med tidigare bilder för att identifiera korrigeringar. I dagsläget hade vi dock klarat oss med ett minneskort som kostar ungefär hälften så mycket. Allt detta leder till en produkt som är väldigt attraktiv ur ett kostnadsperspektiv. En jämförelse har gjorts med två konkurrerande system: powerpointpresentationer och interaktiva whiteboards.

I dagsläget använder många föreläsare powerpointpresentationer som stöd för att

presentera sitt material. Powerpointpresentationerna laddas vanligtvis upp på kursens egna sida i PingPong så att studenterna ges tillgång till dem i efterhand. Om en föreläsare parallellt med presentationen skriver information på tavlan får studenten inte tillgång till informationen från tavlan i efterhand. E-blackboard är designad för att spara denna information och kan därför ses som ett komplement till presentationen. Powerpointpresentationer har fördelar i att de kan färdigställas i förtid samt att de kan redigeras i efterhand om presentationen är felaktig.

En interaktiv whiteboard är en tryckkänslig skärm som monteras upp på en vägg [50]. På denna skärm projiceras en bild från en projektor som kopplas till samma dator som skärmen. Varje tryck och gest på skärmen skickas till datorn som projicerar en ny bild med hjälp av den nya informationen. Detta system sparar den presenterade informationen digitalt från början och systemet erbjuder även en mängd extra funktionalitet. Nackdelen är att det måste finnas en projektor för varje skärm och varje uppsättning är relativt dyr. Många föreläsare uppskattar möjligheten att kunna arbeta med flera tavlor och även kunna referera tillbaka till något som skrevs tidigare under föreläsningen. Att arbeta med fler tavlor är något som skulle bli mycket svårare att göra om endast interaktiva whiteboards användes och skulle därmed ha märkbar inverkan på hur föreläsningar läggs upp.

5.4 E-blackboard och studieteknik

När det handlar om att ta till sig ny kunskap är förståelse A och O, och därför säger det sig självt att när en student går på en föreläsning är det absolut kritiskt att förstå vad föreläsaren vill förmedla. Hur detta uppnås är olika från person till person och beror även på vad för något föreläsningen behandlar, men något som är sant för alla individer är att koncentrationen måste hållas uppe. För de flesta hjälps inlärningen av att anteckna, då antecknandet tvingar personen till att aktivt lyssna och engagera sig [51]. Om inga anteckningar förs är det lätt att tappa koncentrationen och börja tänka på saker som inte är relevant för föreläsningen. Problem uppstår dock om själva aktiviteten att anteckna vad som skrivs på tavlan kräver så pass mycket uppmärksamhet att det är omöjligt att bearbeta den information som föreläsaren förmedlar verbalt. Detta är ett speciellt märkbart fenomen under föreläsningar då det är komplicerade uträkningar som presenteras på tavlan samtidigt som föreläsaren förklarar vad som sker.

Efter att ha suttit igenom en föreläsning har hjärnan tagit in ny kunskap. Den nya kunskapen är dock vag och måste underhållas inom relativt kort tid för att inte förloras. Därför är det viktigt att repetera vad föreläsningen behandlade och på så sätt befästa den nya kunskapen i minnet. Om innehållet dessutom repeteras

ännu en gång inom en vecka kommer kunskapen att kommas ihåg en lång tid framöver.

Det är ovanstående reflektioner som lett till skapandet av E-blackboard. Med E-blackboard kan studenten fokusera på vad som sägs under föreläsningen och föra anteckningar som inte är allt för detaljerade, vilket hjälper både förståelse och att hålla koncentrationen uppe. Studenten har sedan tillförlitlig tillgång till vad som skrevs på tavlorna i efterhand och kan repetera föreläsningens innehåll, något som underlättar inlärningen.

Tyvärr finns det alltid risker för negativa effekter, och så även med E-blackboard. Det finns en risk att studenter lockas till att inte anteckna alls under föreläsningar vilket kan leda till en negativ inverkan på inlärningen. Detta är dock ett val som lämnas upp till den enskilde individen och det finns inget som hindrar att egna anteckningar förs trots att systemet används under föreläsningen.

6

Slutsats

Projektets syfte var att underlätta studenters studier genom att utveckla ett system som elektroniskt tillhandahåller information som presenteras på krittavlor utan att studenter behöver anteckna. Genom att en fungerande prototyp har producerats har vi lyckats med projektets huvudsakliga mål. Att vissa delmål inte uppfyllts till 100 % beror huvudsakligen på att systemet är beroende av interaktion med en föreläsare och därmed kan fel inträffa. Om alla föreläsare instrueras och sedan använder systemet på rätt sätt blir det pålitligt och delmålen, att all information som skrivs ner på tavlan ska digitaliseras samt att enbart tavlan ska synas på bilden, blir uppfyllda.

Att E-blackboard underlättar studenters inlärning anser vi vara uppfyllt. Framförallt ger det studenten bättre förutsättningar att repetera efter en föreläsning vilket leder till att den nya kunskapen kan behållas under en längre tid. Att systemet skulle medföra att fler studenter blir lata och struntar i att föra egna anteckningar är möjligt men det är trots allt upp till var och en att ta ansvar för sina egna studier.

Efter projektets slutförande finns det en högst relevant fråga att ställa: "Är det aktuellt att investera i E-blackboard?". Det är gruppens enade åsikt att systemet i dess aktuella utförande inte är redo för användning i någon större skala. Däremot är systemet på god väg till något som skulle kunna vara aktuellt att använda. Efter att de problem som presenterats i kapitel 5 Diskussion är åtgärdade kan det vara aktuellt att köra ett längre test. Detta test skulle vara att använda systemet för en handfull kurser, där föreläsarna introduceras till hur det fungerar och installera

systemet i de föreläsningssalar som används för de aktuella kurserna. Om systemet då visar sig tillräckligt stabilt och studenterna anser att det hjälper dem i sina studier kan det vara aktuellt att installera E-blackboard i en större skala.

Källförteckning

[1] K. Johansson, "Skanner", *Nationalencyklopedin*, 2014. [Online]. Tillgängling: http://www.ne.se/lang/skanner, [Hämtad: 2014-05-16].

[2] W. Zeng, "Multimedia at work", *IEEE Computer Society*, 2012. [Online]. Tillgängling: http://ieeexplore.ieee.org/ielx5/93/6190801/06190806.pdf?tp=&arnumber=6190806&isnumber=6190801, [Hämtad: 2014-05-16].

[3] Microsoft, *Kinect för xbox 360*. [Online]. Tillgängling: http://www.microsoftstore.com/store/mseea/sv_SE/pdp/Kinect-f%C3%83%C2%B6r-Xbox-360/productID.256402000, [Hämtad: 2014-05-18].

[4] M. Perenson, "Eye-fi share video 4gb sd card with wi-fi", *PCWorld Australia*, 2009. [Online]. Tillgängling: http://www.pcworld.idg.com.au/review/eye-fi/share_video_4gb/306770/, [Hämtad: 2014-05-16].

[5] Github, *Features*, 2014. [Online]. Tillgängling: https://github.com/features, [Hämtad: 2014-06-05].

[6] A. Alliance, *The agile manifesto*, 2014. [Online]. Tillgängling: http://www.agilealliance.org/the-alliance/the-agile-manifesto/, [Hämtad: 2014-06-05].

[7] Scrum.org, *Scrum*, 2014. [Online]. Tillgängling: https://www.scrum.org, [Hämtad: 2014-06-05].

[8] Trello, *Trello*, 2014. [Online]. Tillgängling: https://trello.com, [Hämtad: 2014-06-05].

[9] C. Atwell, *The biggest-little revolution: 10 single-board computers for under $100*, 2013. [Online]. Tillgängling: `http://www.eetimes.com/document.asp?doc_id=1319262&page_number=1`, [Hämtad: 2014-05-18].

[10] R. P. Foundation, *Model a*. [Online]. Tillgängling: `http://www.raspberrypi.org/product/model-a/`, [Hämtad: 2014-05-18].

[11] ——, *Model b*. [Online]. Tillgängling: `http://www.raspberrypi.org/product/model-b/`, [Hämtad: 2014-05-18].

[12] N. Owano, "Raspberry pi gets customized os called raspbian", *Phys.org*, 2012. [Online]. Tillgängling: `http://phys.org/news/2012-07-raspberry-pi-customized-os-raspbian.html`, [Hämtad: 2014-05-10].

[13] Elinux, *Rpi low-level peripherals*, 2014. [Online]. Tillgängling: `http://elinux.org/RPi_Low-level_peripherals#References`, [Hämtad: 2014-05-18].

[14] Adafruit, *Raspberry pi camera board*, 2014. [Online]. Tillgängling: `http://www.adafruit.com/products/1367`, [Hämtad: 2014-05-18].

[15] K. Kalantar-Zadeh, *Sensors: an introductory course*. New York: Springer, 2013.

[16] W. Judd, "An introduction to cherry mx mechanical switches", *The Keyboard Company*, 2012. [Online]. Tillgängling: `http://www.keyboardco.com/blog/index.php/2012/12/an-introduction-to-cherry-mx-mechanical-switches/`, [Hämtad: 2014-05-03].

[17] P. Jain, "Infrared sensors or ir sensors", *Engineers Garage*, 2012. [Online]. Tillgängling: `http://www.engineersgarage.com/articles/infrared-sensors?page=2`, [Hämtad: 2014-05-03].

[18] E. Ramsde, *Hall-effect sensors: theory and applications, second edition*. Boston: Elsevier, 2006.

[19] Webbhotellfakta, *Olika typer av hosting*, 2011. [Online]. Tillgängling: `http://www.webbhotellfakta.se/hosting.aspx`, [Hämtad: 2014-04-13].

[20] P. Whitehead och J. Russell, *HTML : your visual blueprint for designing Web pages with HTML, CSS, and XHTML*. Hoboken, N.J.: Wiley, 2005, s. 18–34.

[21] Contentmanager, *What is web content management*, 2012. [Online]. Tillgängling: `http://www.contentmanager.eu.com/wcms.htm`, [Hämtad: 2014-05-13].

[22] K. Nilsson, "Databas", *Nationalencyklopedin*, 2014. [Online]. Tillgängling: `http://www.ne.se/lang/databas`, [Hämtad: 2014-05-16].

[23] J. Błażewicz, W. Kubiak, T. Morzy och M. Rusinkiewicz, *Handbook on Data Management in Information Systems*. Berlin: Springer, 2003.

[24] K. Grolinger, W. Higashino, A. Tiwari och M. Capretz, "Data management in cloud environments: nosql and newsql data stores", *Journal of Cloud Computing*, 2013. [Online]. Tillgängling: `http://www.journalofcloudcomputing.com/content/pdf/2192-113X-2-22.pdf`, [Hämtad: 2014-05-16].

[25] E. Rosebrock och E. Filson, *Setting up LAMP : getting Linux, Apache, MySQL and PHP working together*. 2004.

[26] M. Kofler och D. Kramer, *MySQL*. Berkeley, Calif: Apress, 2001, s. 4–19.

[27] G. Allen och M. Owens, *The Definitive Guide to SQLite*. Berkeley, Calif: Apress, 2010, s. 1–16.

[28] W. Goralski, "Chapter 25 - secure shell (remote access)", i *The Illustrated Network*, W. Goralski, utg., Boston: Morgan Kaufmann, 2009, s. 633–658.

[29] R. Szeliski, *Computer Vision*, English. Springer London, 2011, s. 1–25.

[30] K. Demaagd, A. Oliver, N. Oostendorp och K. Scott, *Practical Computer Vision with SimpleCV*. Sebastopol: O'Reilly Media, 2012, s. 1–7.

[31] A. Mordvintsev och K. Abid, "Canny edge detection", *Read the Docs*, 2014. [Online]. Tillgängling: `http://opencv-python-tutroals.readthedocs.org/en/latest/py_tutorials/py_imgproc/py_canny/py_canny.html`, [Hämtad: 2014-05-16].

[32] K. Demaagd, A. Oliver, N. Oostendorp och K. Scott, *Practical Computer Vision with SimpleCV*. Sebastopol: O'Reilly Media, 2012, s. 149–170.

[33] N. Otsu, "A threshold selection method from gray-level histograms", *IEEE transactions on systems, man, and cybernetics*, vol. 9, nr 1, s. 62–66, 1979.

[34] D. Polka, *Chapter 3: ac and dc motors - servomotors: general principles of operation*, 2003. [Online]. Tillgängling: `http://www.globalspec.com/reference/10801/179909/chapter-3-ac-and-dc-motors-servomotors-general-principles-of-operation`, [Hämtad: 2014-05-19].

[35] B. Somnath, "Raspberry pi: controlling eight servo motors", *Electronics for You*, 2013. [Online]. Tillgängling: `http://proxy.lib.chalmers.se/login?url=http://search.proquest.com/docview/1519071113?accountid=10041`, [Hämtad: 2014-05-18].

[36] Mechanicalkeyboards, *Cherry mx brown keyswitch - plate mount - tactile - 5 pack (cherry)*, 2014. [Online]. Tillgängling: `http://mechanicalkeyboards.com/shop/index.php?l=product_detail&p=708`, [Hämtad: 2014-05-18].

[37] Binero, *Jämför webbhotell och priser*, 2014. [Online]. Tillgängling: `http://www.binero.se/webbhotell/jamfor-pris`, [Hämtad: 2014-05-19].

[38] WordPress, *Learn wordpress*, 2014. [Online]. Tillgängling: `http://learn.wordpress.org/`, [Hämtad: 2014-05-18].

[39] Bootstrap, *Getting started*, 2014. [Online]. Tillgängling: `http://getbootstrap.com/2.3.2/getting-started.html`, [Hämtad: 2014-05-18].

[40] Formstone, *Zoomer*, 2014. [Online]. Tillgängling: `http://formstone.it/components/zoomer`, [Hämtad: 2014-05-18].

[41] J. Tidwell, *Designing Interfaces 2:nd edition*. Sebastopol: O'Reilly, 2011, s. 77–191.

[42] K. Demaagd, A. Oliver, N. Oostendorp och K. Scott, *Practical Computer Vision with SimpleCV*. Sebastopol: O'Reilly Media, 2012, s. 81–97.

[43] ——, *Practical Computer Vision with SimpleCV*. Sebastopol: O'Reilly Media, 2012, s. 51–78.

[44] S. M. Inc, *Api*, 2014. [Online]. Tillgängling: `http://simplecv.org/docs/SimpleCV.html#i/SimpleCV.ImageClass.Image/floodFill`.

[45] J. Illingworth och J. Kittler, "A survey of the hough transform", *Computer Vision, Graphics, and Image Processing*, vol. 44, nr 1, s. 87–116, 1988. [Online]. Tillgängling: `http://www.sciencedirect.com/science/article/pii/S0734189X88800331`.

[46] D. J. C. MacKay, *Information theory, inference, and learning algorithms*. Cambridge, UK; New York: Cambridge University Press, 2003, s. 284–292.

[47] F. Lundh, *Python imaging library handbook*, 2014. [Online]. Tillgängling: `http://effbot.org/imagingbook/imagefilter.htm`.

[48] ——, *Python imaging library handbook*, 2014. [Online]. Tillgängling: `http://effbot.org/imagingbook/pil-index.htm`.

[49] C. Clapham och J. Nicholson, *The Concine Oxford Dictionary of Mathematics*. Oxford University Press, 2009.

[50] Smarttech, *Smart board m600 interactive whiteboard*, 2013. [Online]. Tillgängling: `http://downloads01.smarttech.com/media/sitecore/en/pdf/products/ifp/sbm600factsheet.pdf`.

[51] B. Liljeqvist, *Plugga smart och lär dig mer!* Lund: Studentlitteratur, 2006, s. 15–31.

A

Projektkostnader

Tabell A.1: *Verkliga samt beräknade kostnader för projektet.*

Produkt	**Verklig Kostnad [Kr]**	**Beräknad kostnad [Kr]**
Raspberry Pi	242,78	242,78
SD-minneskort	225,92	225,92
Camera board	170,09	170,09
Hölje Raspberry Pi	52,32	52,32
Tower Pro SG90	99,90	99,90
Cherry MX Brown	Gratis	45,40 (1 USD = 6,49 SEK)
Lapp kabel	Gratis	49,84 (Beräknat pris från 100m)
Summa:	**791,01**	**886,25**

B

Testdata från systemtest

Tabell B.1: *Test 1: grundläggande funktioner.*

Funktion som testas	**Resultat**
Automatisk uppstart	Lyckades
Kamera tar bild på given sensorsignal	Lyckades
Bild tagen av kameran ger läsbar text	Lyckades
Båda tavlor kommer med på samma bild när enheten är placerad på rimligt avstånd	Misslyckades
Bilder laddas upp till webbplatsen och går att nå från gränssnittet	Lyckades

Tabell B.2: *Test 2: Bildbehandling.*

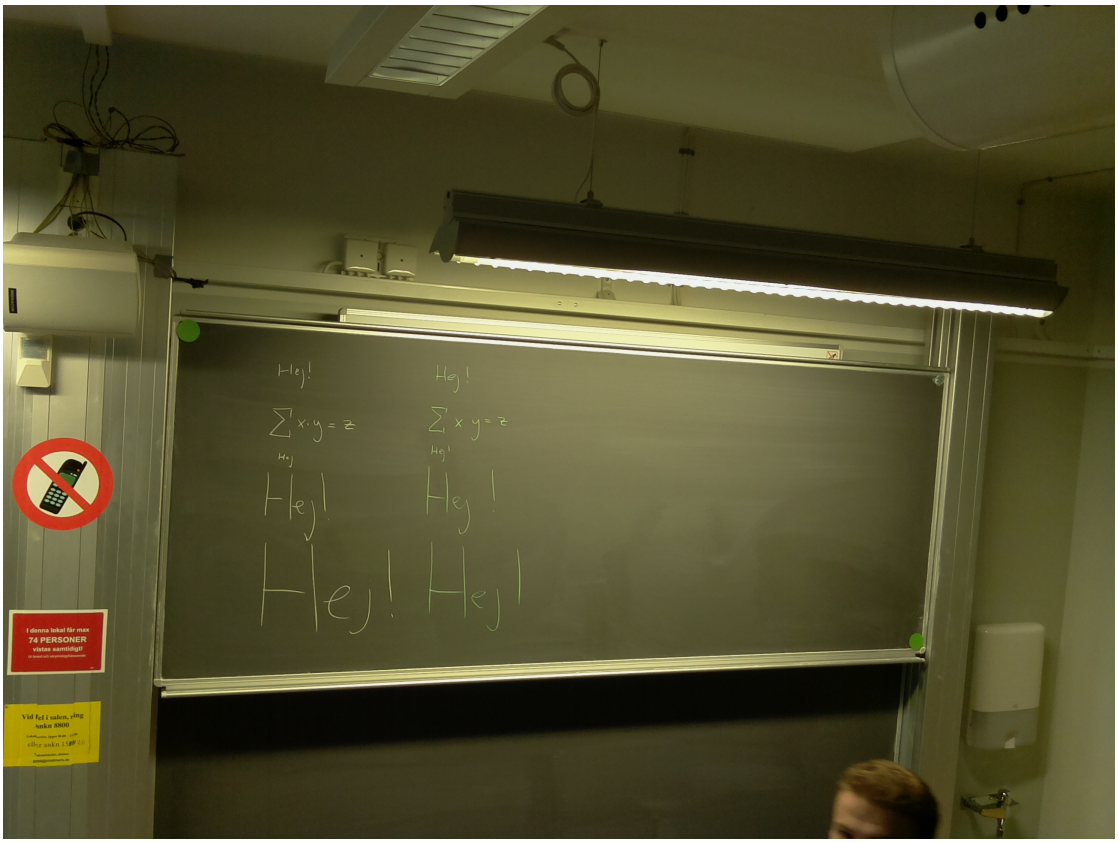
Funktion som testas	**Resultat**
Bild går vidare till bildbehandlingsprogrammet	Lyckades
Tavla hittas och klipps ut efter sensorsignal	Misslyckades delvis. Osäkert resultat, lyckades ca 1/4 av försöken.

Tabell B.3: *Test 3: slutgiltigt test.*

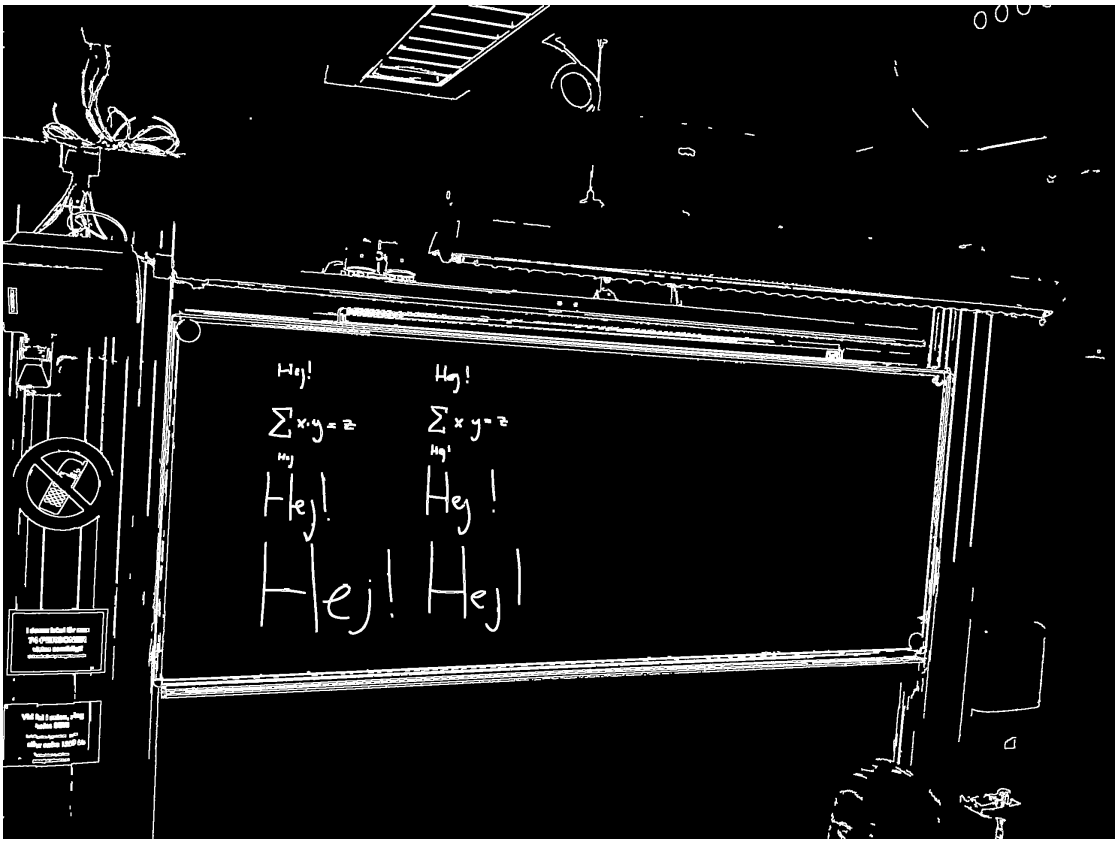
Funktion som testas	**Resultat**
Motor styr enheten till korrekt tavla efter sensorsignal	Lyckades
Tavla hittas och klipps ut efter motorstyrnig och bildtagning	Lyckades
Tavla hittas och klipps om någon står 0.5 meter framför tavlan	Lyckades
Tavla hittas och klips ut om någon står precis intill tavlan och skymmer sikten	Misslyckades

C

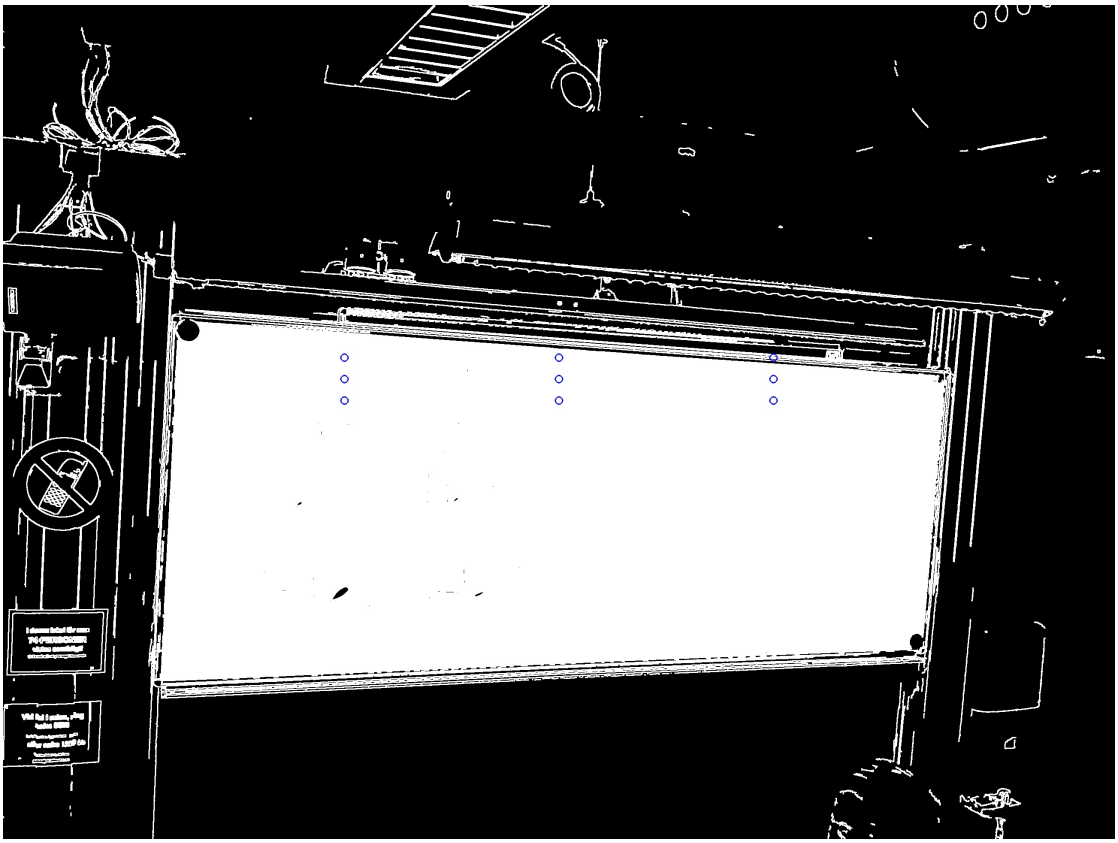
Bildredigeringsprocess



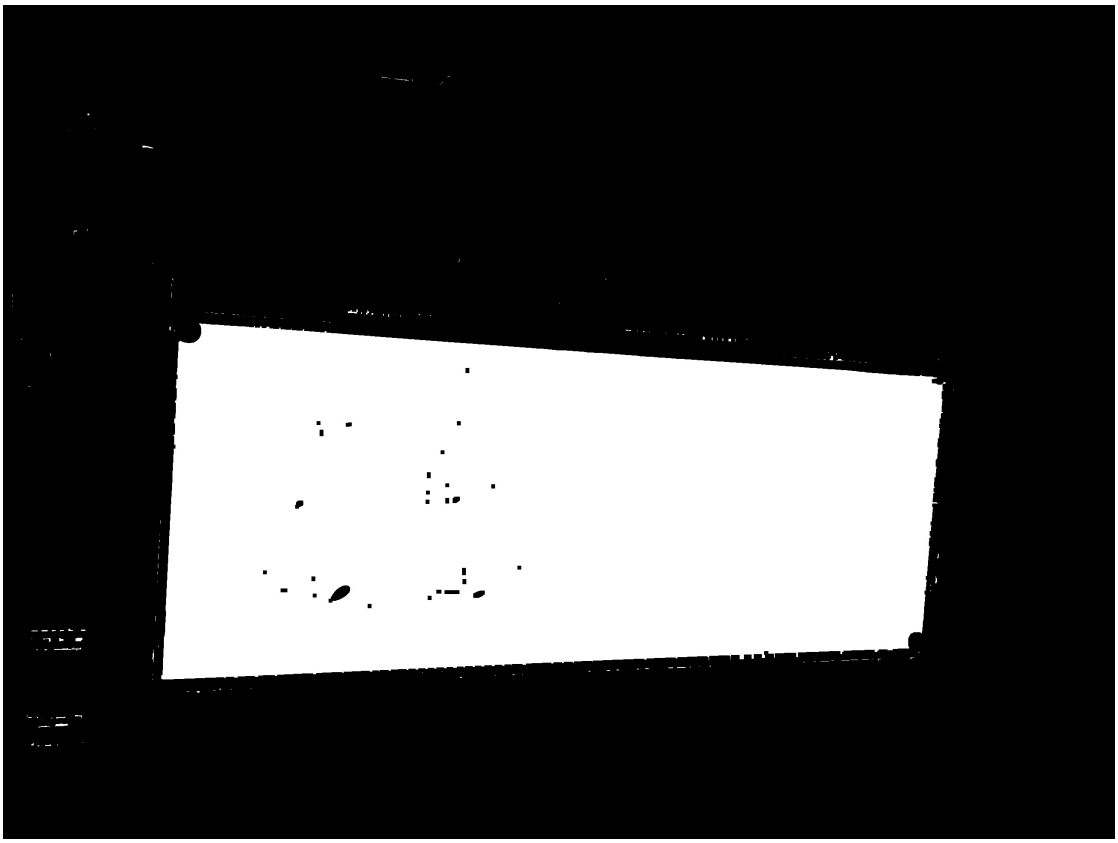
Figur C.1: *Den ursprungliga bilden som ska processeras*



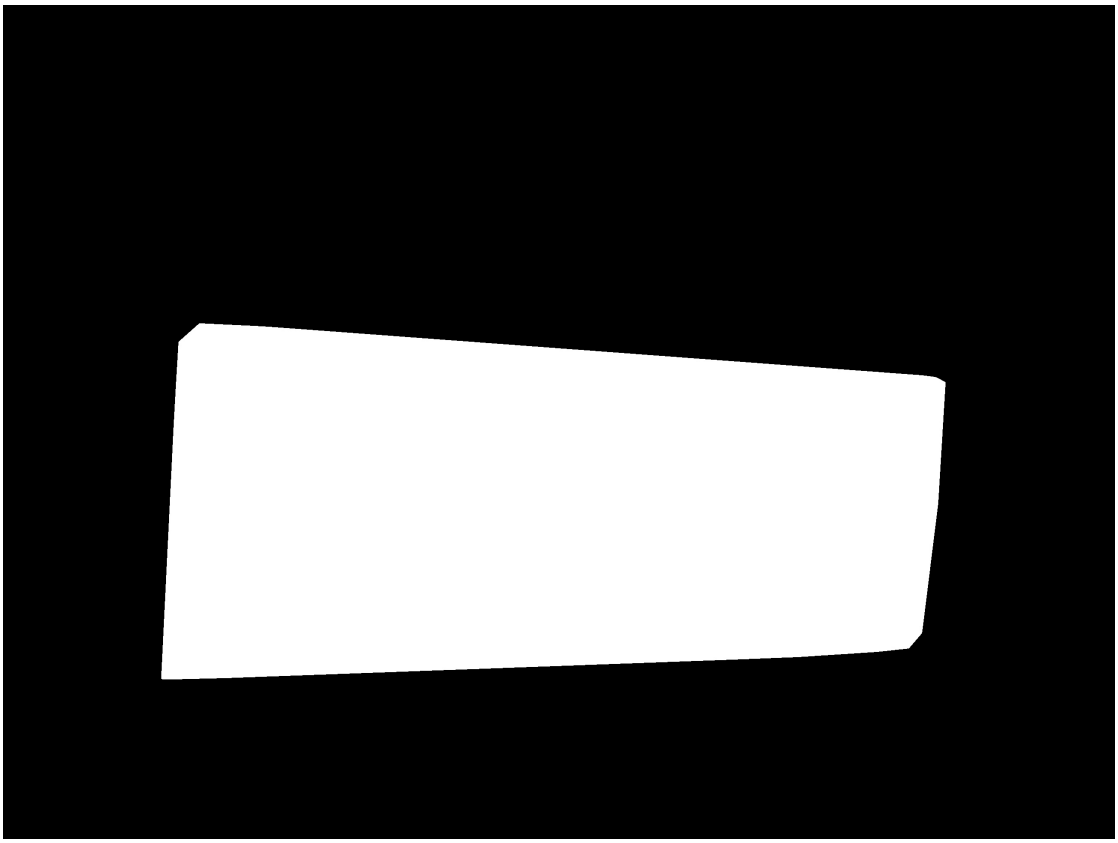
Figur C.2: *Bildens utseende efter Canny kantdetektering och förstoring av vita pixlar*



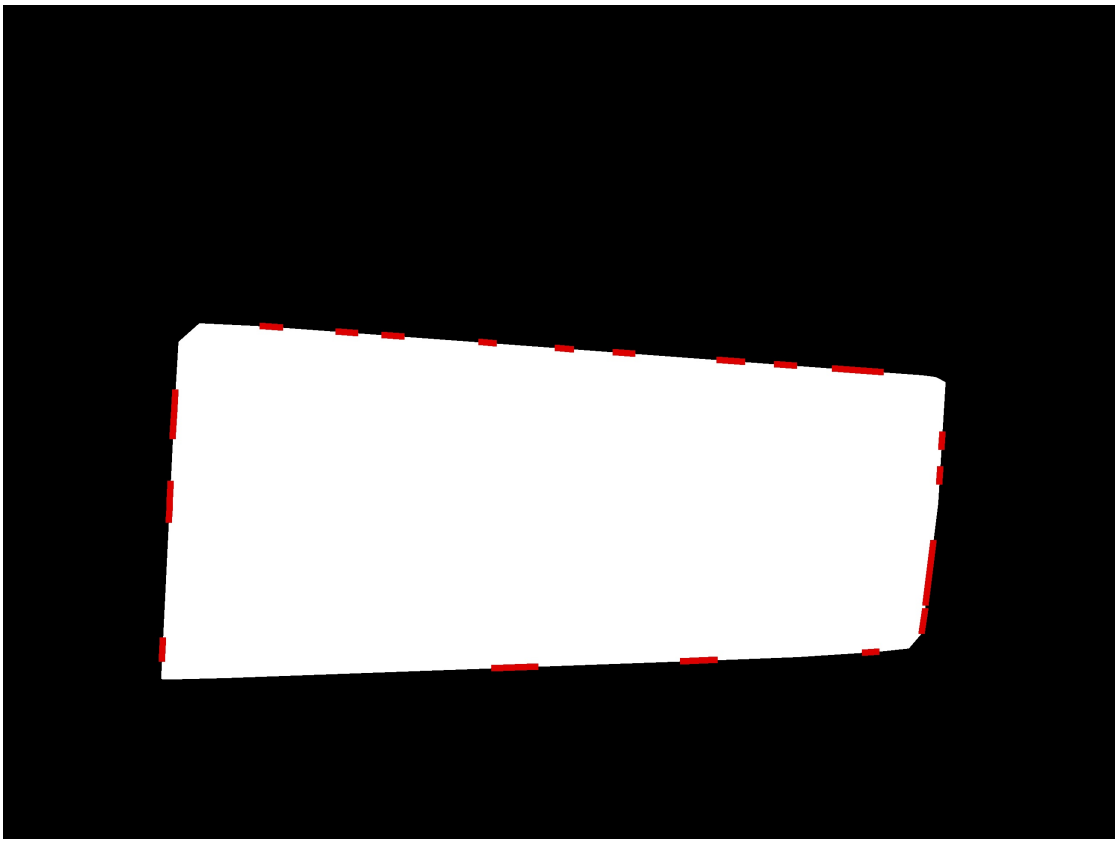
Figur C.3: *Resultatet av att floodFill körts med markerade utgångspunkter*



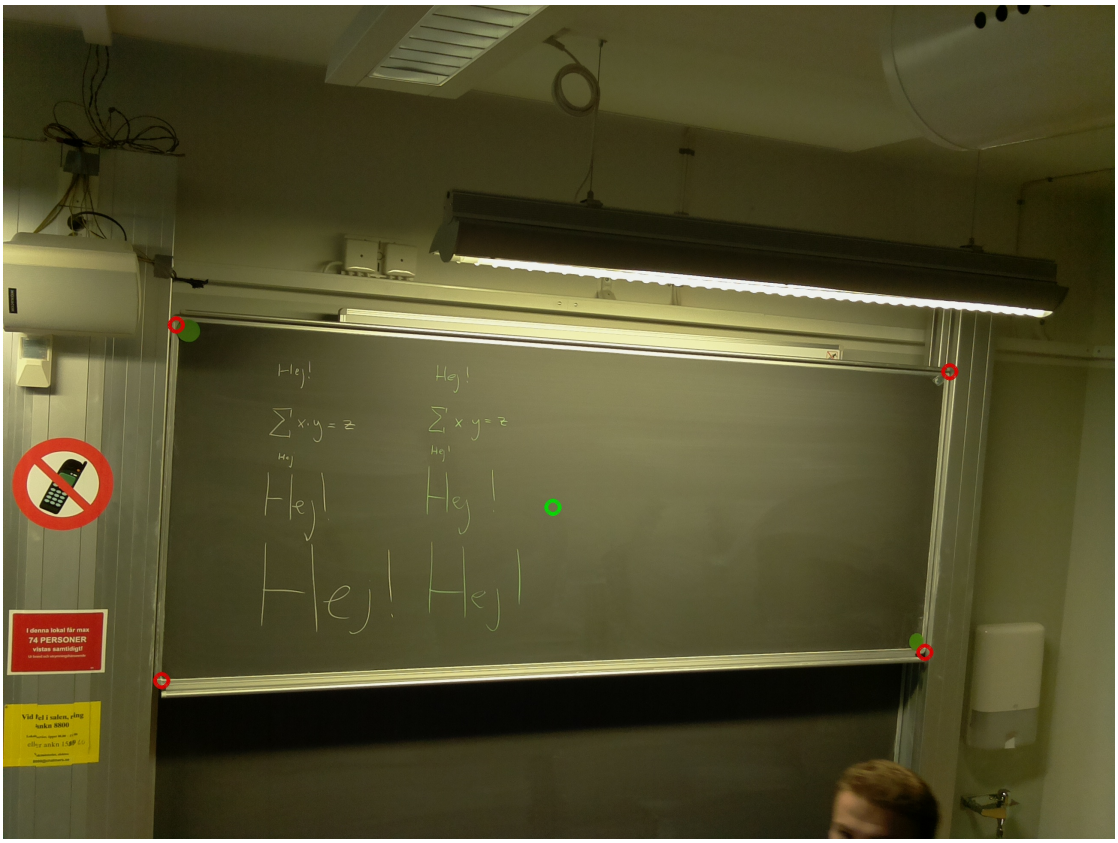
Figur C.4: *Resultatet av Erode funktionen. Majoriteten av alla tunna linjer har försvunnit och kvar är majoriteten av tavlan.*



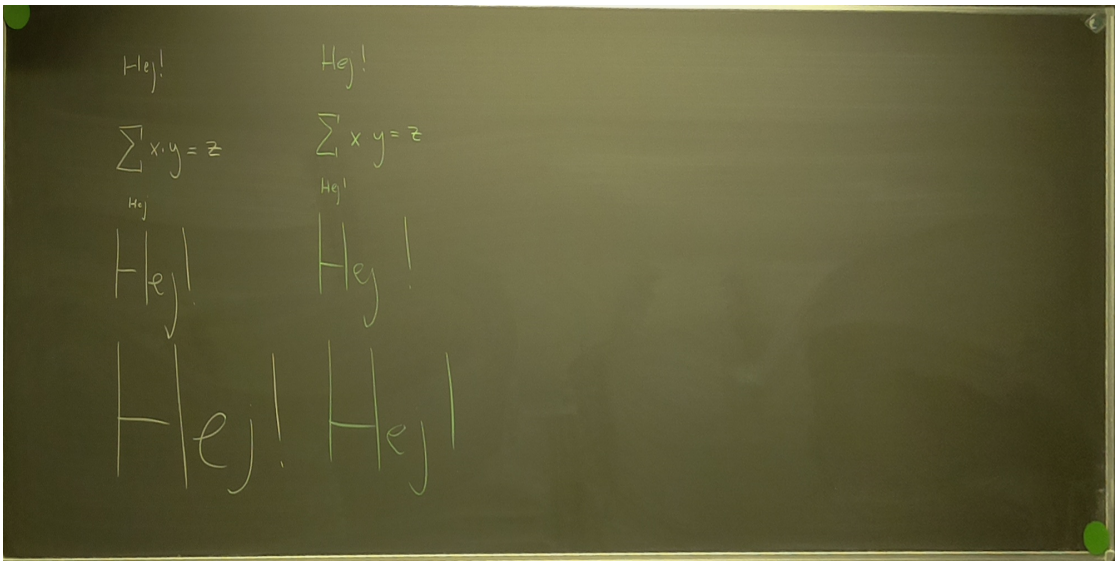
Figur C.5: *Den konvexa masken som skapas för att kunna fortsätta arbeta med bilden*



Figur C.6: *Masken med funna linjer från Hough transformen.*



Figur C.7: *Ursprungsbilden med beräknade hörnpositioner markerade tillsammans med den projicerade tavlans mittpunkt.*



Figur C.8: *Hörnen används för att beräkna en rotationsmatris som korrigerar perspektivet och sedan klipps tavlan ut för att bara ha med den intressanta informationen.*

D

Mätdata: Undersökning av energiförbrukning

Tabell D.1: *Mätvärden från ström och spänningsmätningen.*

Tid [s]	Ström [A]	Spänning [V]	Effekt [W]
0	0	0	0
5	0,13	4,90	0,64
10	0,27	5,01	1,35
15	0,23	5,00	1,15
20	0,32	4,98	1,59
25	0,30	5,00	1,50
30	0,27	5,02	1,36
35	0,35	5,00	1,75
40	0,37	5,00	1,85
45	0,40	4,98	1,99
50	0,42	5,02	2,11
55	0,43	5,00	2,15
60	0,46	5,00	2,30
65	0,46	5,01	2,30
70	0,46	5,00	2,30
75	0,46	5,00	2,30
80	0,46	5,01	2,30
85	0,46	5,01	2,30
90	0,46	5,00	2,30
95	0,49	4,98	2,44
100	0,53	5,00	2,65
105	0,51	5,01	2,56
110	0,60	5,02	3,01
115	0,62	5,00	3,10
120	0,63	5,00	3,15
125	0,51	4,98	2,54
130	0,46	5,00	2,30
135	0,46	5,01	2,30
140	0,45	5,00	2,25

145	0,46	5,00	2,30
150	0,46	5,00	2,30
155	0,49	5,00	2,45
160	0,52	5,00	2,60
165	0,55	4,98	2,74
170	0,60	5,00	3,00
175	0,65	5,00	3,25
180	0,60	5,02	3,01
185	0,55	5,00	2,75
190	0,47	5,00	2,35
195	0,46	5,00	2,30
200	0,45	5,01	2,25
205	0,46	5,00	2,30
210	0,46	5,00	2,30
215	0,47	5,00	2,35
220	0,49	5,00	2,45
225	0,51	5,00	2,55
230	0,58	4,99	2,89
235	0,63	5,00	3,15
240	0,64	5,00	3,20
245	0,62	5,00	3,10
250	0,57	5,02	2,86
255	0,48	5,00	2,40
260	0,46	5,00	2,30
265	0,46	5,00	2,30
270	0,46	5,03	2,31
275	0,45	5,00	2,25
280	0,47	5,00	2,35
285	0,49	4,98	2,44
290	0,55	4,99	2,74
295	0,59	5,00	2,95

300	0,62	5,00	3,10
305	0,59	5,00	2,95
310	0,56	5,00	2,80
315	0,47	5,00	2,35
320	0,46	5,00	2,30
325	0,47	5,00	2,35
330	0,46	5,00	2,30
335	0,22	5,00	1,10
340	0,16	5,00	0,80
345	0,00	0	0

E

Matlabkod

I den här bilagan presenteras matlabkoden som användes i undersökningen av enhetens energiförbrukning.

E.1 Energikostnaden

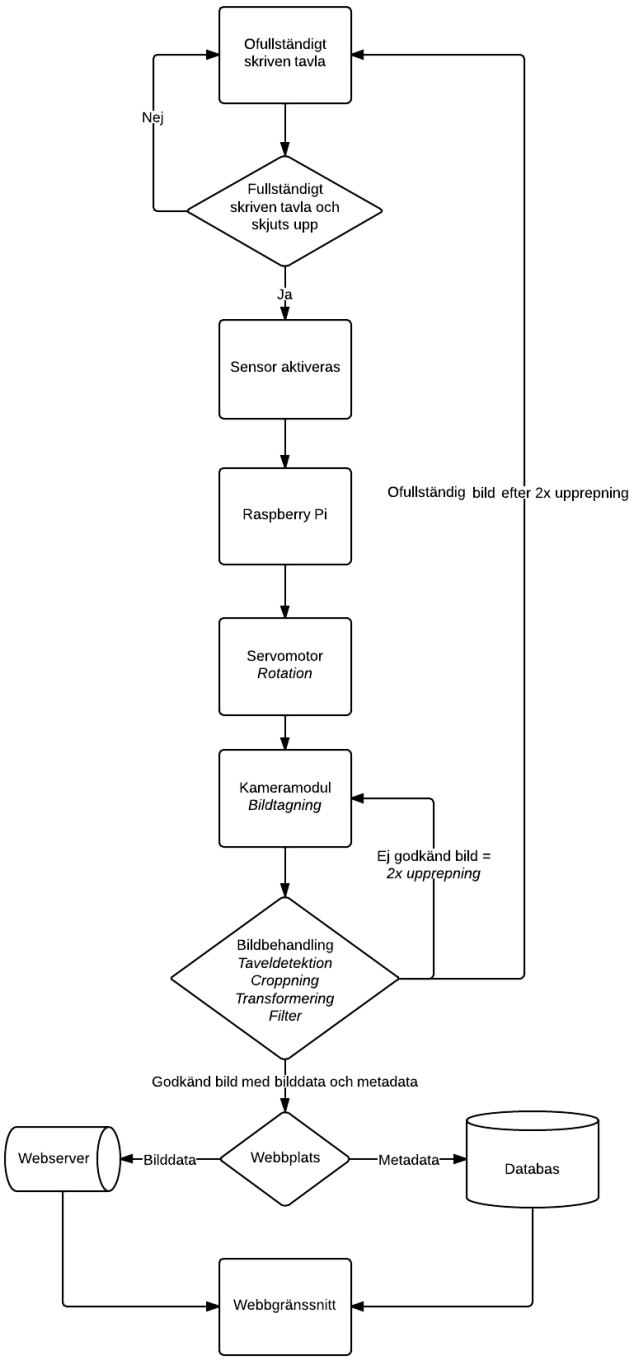
```
%Beräkningar på energikostnaden

Z = cumtrapz(Tidsek/3600,WattW);  %Omvandlar mätvärdena till kWh
Zny = abs(Z)/1000;  %Omvandlar mätvärdena till kWh

%Summera ihop energin för de olika aktiviterna

SummaUppstart = Zny(15);
SummaSensorA = Zny(27)-Zny(19);
SummaSensorB = Zny(39)-Zny(31);
SummaSensorC = Zny(52)-Zny(42);
SummaSensorD = Zny(64)-Zny(56);
SummaNedstangning = Zny(70)-Zny(67);
SummaVantelage= (Zny(19)-Zny(15)) + (Zny(31)-Zny(27)) + (Zny(44)-Zny(39))
+ (Zny(56)-Zny(52)) + (Zny(67)-Zny(64));
MedelvardeSensorer = (SummaSensorA + SummaSensorB + SummaSensorC +
SummaSensorD)/4;
```

E.2 Daglig energiförbrukning

```
%Uträkningar på energiförbrukningen under en dag

AktiveradSensor = 15*4*MedelvardeSensorer;
SummaUnderDagen = AktiveradSensor + 0.0203 + SummaUppstart + Summa-
Nedstangning;

%Beräkningarn på energiförbrukningen under ett år av föreläsningar

PrisetUnderDagen = SummaUnderDagen*171*0.8038;
```

F

Flödesdiagram över en processcykel



Figur F.1: *Flödesdiagram från det att en fullständigt skriven tavla skjuts upp till att en färdig bild laddas upp på hemsidan.*