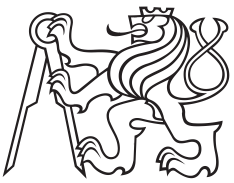
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra kybernetiky

Čtení čárových kódů kamerou

Maksim Kiselev
Studijní program: Kybernetika a robotika(bakalářsky)
Obor: Robotika

květen 2017
Vedoucí práce: Ing. Pavel Krsek, Ph.D

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra kybernetiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: Maksim K i s e l e v

Studijní program: Kybernetika a robotika (bakalářský)

Obor: Robotika

Název tématu: Čtení čárových kódů kamerou

Pokyny pro vypracování:

1. Seznamte se se standardy definovanými pro čárové kódy a připravte jejich přehled. Zaměřte se především na 1D kódy (EAN, JAN, UPC, Interleaved 2 of 5 any length) a jejich rozměry.
2. Stanovte teoretické hranice pro rozlišení obrazu, v němž bude možné kódy číst.
3. Vyhledejte stávající řešení (knihovny, programy) pro čtení čárových kódů v obraze pořízeném kamerou. Cílovou platformou je mobilní zařízení (brýle virtuální reality či mobilní telefon).
4. Posuďte jednotlivá řešení a vyberte několik z nich pro implementaci a testování.
5. Připravte data pro testování čtení čárových kódů kamerou. Data připravte s ohledem na testování minimálního rozlišení a kontrastu nutného pro úspěšné čtení kódů.
6. Implementovaná řešení otestujte a připravte přehled s doporučením pro další použití. Součástí testování by mělo být i orientační měření výpočetní náročnosti.

Seznam odborné literatury:

[1] Milan Sonka, Vaclav Hlavac, and Roger Boyle. Image Processing, Analysis and Machine Vision. Thomson, 3rd edition, ISBN 978-0-495-08252, 2007.
[2] GS1 (www.gs1.org), Version 15, GS1 General Specifications, GS1, January 2015 http://www.gs1.org/sites/default/files/docs/barcodes/GS1_General_Specifications.pdf

Vedoucí bakalářské práce: Ing. Pavel Krsek, Ph.D.

Platnost zadání: do konce letního semestru 2016/2017

L.S.

prof. Dr. Ing. Jan Kybic
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 15. 12. 2015

Poděkování / Prohlášení

Chtěl bych poděkovat svému vedoucímu pánu Krsku za pomoc v jazykovém zpracování dokumentu a teoretickou podporu provedených experimentů a taky rodičům za morální a finanční podporu.

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 26.05.2017

.....

Abstrakt / Abstract

Tento dokument obsahuje výsledky experimentů nad SDK pro načítání čárových kódů. Experimenty měli za cíl vyzkoušet možností (přesnost a rychlost) a, případně, stanovit závislost načítání čárových kódů pomocí nalezených SDK při různých vzdalovanostech, uhlech natočení čárových kódů a úrovních osvětlení.

Klíčová slova: čárové kódy; načítání čárových kódů; charakteristiky čárových kódů; bakalářská práce; maximální vzdálenost načítání čárových kódů.

This document contains the results of the experiments over the barcode reading SDKs. The experiments were aimed to test the characteristics (accuracy and speed) and to determine the mathematical dependency of barcode reading SDKs depending on different distances, angles of rotation of barcodes, and lighting levels.

Keywords: barcodes; barcodes reading; barcodes parameters; Bachelor project; maximal distance of barcode reading.

Title translation: Barcode reading with camera

Obsah /

Tabulky / Obrázky

Kapitola 1
Úvod

Tento dokument popisuje možnosti načítání čárových kódů pomocí kamery. Práce byla oddělena od projektu pro integrací rozšířené reality na sklad, a proto je zaměřená na malé kamery s velkým úhlem pohledu (stejné jaké jsou instalovány do mobilů). Základním cílem je najít různé SDK pro načítání čárových kódů, stanovit praktické meze jejich načítání, hustota pixelu na unit pro úspěšné přečtení, potřebný kontrast čárového kódu. Charakteristiky podle kterých byli hodnoceny SDK: čas potřebný pro přečtení kódů, spolehlivost načítání kódů, možnosti přečtení kódů pří různých úhlech, možnosti načítání kódů při malem úrovní kontrastu.

1.1 Seznám použité terminologie a zkratek

- Unit – nejtenčí možná čára v čárovém kódu, představující nejmenší jednotku informací.
- Manatee – zkrátka použita v tabulkách a textech pro ManateeWorks SDK.
- Scandit – zkrátka použita v tabulkách a textech pro Scandit SDK.
- ZBar – ZBar bar code reader.
- ZXing – ZXing (”Zebra Crossing”).
- B – v popisu standardů čárových kódů označuje mezeru.
- Č – v popisu standardů čárových kódů označuje čáru.

Kapitola 2
Teoretický základ experimentů

V této kapitole budeme uvažovat, že máme ideální podmínky pro čtení kódů, tedy ideální horizontální poloha čárového kódu, perfektní záchyt obrázku (bez šumu), vysoká úroveň osvětlení. Z Shannonůva teorému, který zní jako „Aby bylo možné zrekonstruovat původní signál, musí být vzorkovací frekvence alespoň dvakrát větší než nejvyšší frekvence v signálu.“[19] vyplývá, že libovolné SDK by mělo přečíst čárový kód za podmínky, že jeden unit je zobrazen dvěma pixely. V našem případě, pří těchto charakteristikách kamery

Parametry kamery (režim 8.0M):

- Rozměry kamery: 8.5mm x 8.5mm x 5.3mm
- Rozměr aktivní části matici 4365.6μm x 3678.3μm
- Rozlišení: 3264 x 2448
- Vzdálenost matici od čočky: 3557.8μm
- Vertikální úhel pohledu: 31.53^o
- Horizontální úhel pohledu: 24.56^o

vzdálenost, ze které by bylo možné přečíst čárový kód libovolným SDK za nepřítomnosti šumu, může být vypočtená podle formule, která je odvozená za uvažování zjednodušeného modelu kamery (Obr.2.1):

$$\frac{2x_{px}}{f} = \frac{x_{unit}}{D}$$

$$D = \frac{fx_{unit}}{2x_{px}}$$

kde
x_{px} - rozměry(šířka) jednoho pixelu.
f - vzdálenost čočky od matici.
x_{unit} - šířka jednoho unitu.
D - vzdálenost obrázku od kamery.
Při těchto parametrech kamery $D = 43.89cm$

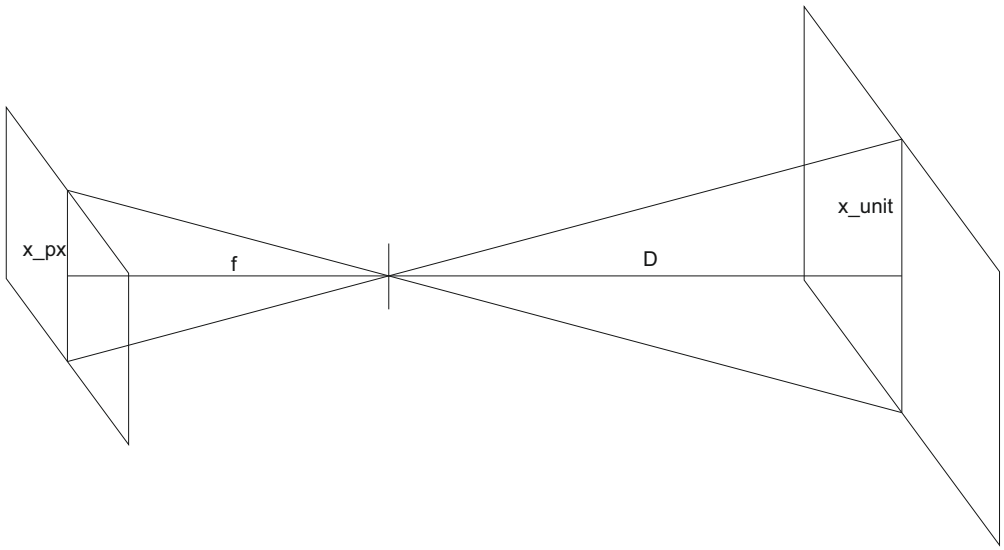
Maximální vzdálenost, ze které by bylo možné přečíst čárový kód za ideálních podmínek, může být vypočtená podle formule, která je odvozená za uvažování zjednodušeného modelu kamery (Obr.2.1):

$$\frac{x_{px}}{f} = \frac{x_{unit}}{D}$$

$$D = \frac{fx_{unit}}{x_{px}}$$

kde
x_{px} - rozměry(šířka) jednoho pixelu

f - vzdálenost čočky od matici.
x_{unit} - šířka jednoho unitu.
D - vzdálenost obrázku od kamery.
Při těchto parametrech kamery $D = 87.78cm$.
Ideálními podmínkami je myšlen ideálně přesný dopad pixelů na unity, tak, aby sloupec pixelů zobrazoval právě jeden unit.



Obrázek 2.1. Zjednodušený model kamery.

2.1 Rotace okolo osy Y

Zavedeme si osovou konvenci pro obrázek2.2. Rotace okolo osy Y způsobí zmenšení viditelného rozměru unitu(nebo rozměru obrázku) a to podle teto rovnice

$$x_{new} = x * cos\alpha$$

kde
α - úhel natočení obrázku.
x_{new} - rozměr unitu po otočení obrázku.
x - rozměr unitu před otočením obrázku.

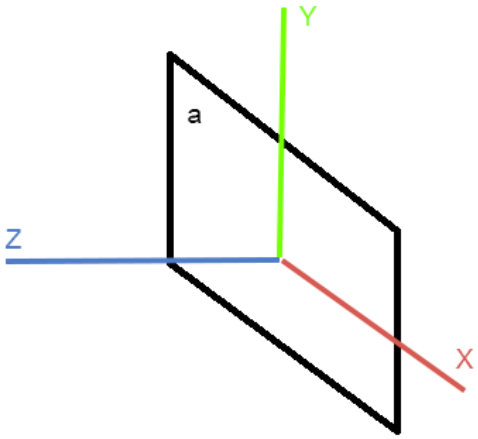
Podle zjednodušeného modelu kamery2.1 rotace okolo osy Y pak způsobí pokles vzdálenosti přečtení kódu v teto proporci

$$\frac{x_{px}}{f} = \frac{x_{unit}}{D}$$

$$\frac{x_{px}}{f} = \frac{x_{unit} * cos\alpha}{D * cos\alpha}$$

$$D_{new} = D * cos\alpha$$

kde
α - úhel natočení obrázku.
x_{px} - rozměry(šířka) jednoho pixelu
f - vzdálenost čočky od matici.
x_{unit} - šířka jednoho unitu.
D - vzdálenost přečtení obrázku před otočením.
D_{new} - vzdálenost přečtení obrázku po otočení.



Obrázek 2.2. Osova konvence obrázku.

Kapitola 3
Nalezené SDK

V této kapitole jsou popsané dostupné na trhu SDK a jejich charakteristiky:

- Podporované formáty kódů
- Dostupnost
- Licence
- Podporované platformy a jazyky

3.1 ZXing

ZXing(Zebra Crossing)[17] je open-source SDK pro načítání různých 1D a 2D kódu. Vyvíjí se v Java pod licencí „Apache 2.0“. Je portována do různých programovacích jazyku (C++, C#, Ruby, Python, JavaScript, PHP, Delphi, Java). SDK je základem pro velké množství bezplatných projektů a nadstaveb. Projekt existuje od 21. října 2007[17](Datum založení github stránky) a stále je aktivní.

Podporované formáty kódů jsou:
UPC-A
UPC-E
EAN-13
EAN-8
Code 39
Code 93
Code 128
Codabar
ITF(Interleaved 2 of 5)

RSS-14
RSS-Expanded

QR Code
Data Matrix
Aztec (beta)
PDF 417 (beta)

3.2 ZBar

Zbar Bar Code Reader[15] je open-source software pro načítání čárových kódů z různých zdrojů jako video streamy, obrázky a senzory. Je vyvíjen v C pod licencí GNU LGPL 2.1. Projekt existuje od 19. únoru 2007[16](datum založení github projektu). Pravděpodobně aktivní podpora a vývoj projektu jsou uzavřený[16](Poslední aktivita

na github je 15. října 2012). SDK zahrnuje podporu pro integrací do Qt, GTK+ PyGTK, podporu perl a python(jsou avšak použité přes wrappery). SDK je integrovatelná taky do iOS a Android.

Podporované formáty kódů jsou:
UPC-A
UPC-E
EAN-13
EAN-8
Code 39
Code 128
Codabar
ITF(Interleaved 2 of 5)

QR Code

3.3 Scandit SDK

Placené SDK. Podporuje velké množství kódů, včetně 2D kódů. Developer říká, že SDK dokáže číst kódy v prostředí s nízkou úrovní osvětlení, pří velkých uhlech pohledu kamery, rozmazané kódy a taky poškozené kódy. SDK je integrovatelné do těchto platforem: iOS, Android, Windows, Linux, Google Glass, Xamarin, PhoneGap / Cordova, Titanium, IBM MobileFirst, hybris, Telerik, Ionic.

Podporované formáty kódů jsou:
UPC-A
UPC-E
EAN-13
EAN-8
Code 39
Code 93
Code 128
Code 25
Code 11
GS1 Databar
MSI plessey
Codabar
ITF(Interleaved 2 of 5)
Standard 2 of 5

RSS-14
RSS-Expanded

QR Code
Data Matrix
Aztec
PDF 417
MicroPDF 417

MaxiCode
GS1 COMPOSITE CODE C (CC-C)
GS1 COMPOSITE CODE B (CC-B)
GS1 COMPOSITE CODE A (CC-A)

3.4 ManateeWorks SDK

Placené SDK. Podporuje velké množství kódů, včetně 2D kódů. Developer říká, že SDK dokáže číst kódy v prostředí s nízkou úrovní osvětlení, pří velkých uhlech pohledu kamery a poškození kódů. SDK je integrovatelné do těchto platforem a frameworků: iOS, MAC OS X, Android, Windows, Windows Phone, Linux, Google Glass, Vuzix, Xamarin, PhoneGap / Cordova, Titanium, IBM MobileFirst, hybris, Telerik, Ionic, iFormBuilder, Meteor, Java.

Podporované formáty kódů jsou:
UPC-A
UPC-E
EAN-13
EAN-8
Code 39
Code 93
Code 128
Code 25
Code 11
GS1 Databar
MSI plessey
Codabar
ITF(Interleaved 2 of 5)
Standard 2 of 5

QR Code
Data Matrix
Aztec
PDF 417
Postal Code
MaxiCode
DotCode

3.5 i-Nigma

Placené SDK. Developer říká, že SDK dokáže číst kódy v prostředí se špatnými podmínkami pro načítáni a pří velkých uhlech pohledu kamery. SDK je integrovatelné do těchto platforem: iPhone (SWIFT and Objecive-C), Android, Windows 10 + Mobile, Windows, iPhone 8, BlackBerry, Windows Mobile / WinCE. SDK nemá testovací licenci.

Podporované formáty kódů jsou:

UPC-A
UPC-E
EAN-13
EAN-8
Code 39
Code 128
GS1 Databar
Codabar
ITF(Interleaved 2 of 5)

QR Code
Micro QR
Data Matrix
PDF 417

3.6 VSBarcodeReader

Placené SDK. Nepodporuje 2D kódy(Existuje VSReaderQR pro načítání QR kódů). Developer říká, že SDK dokáže číst rozmazané kódy. SDK je integrovatelné do Anroid a iOS. SDK nemá testovací licenci.

Podporované formáty kódů jsou:
UPC-A
UPC-E
EAN-13
EAN-8
Code 39
Code 93
Code 128
ITF(Interleaved 2 of 5)
Standard 2 of 5
Codabar

Kapitola 4
Standarty čárových kódů

V této kapitole jsou popsané existující čárové kódy a jejich charakteristiky:

- Sestavení kódu
- Přípustné rozměry
- Způsob vypočtu kontrolního symbolu
- Způsob kódování informací
- Podporované platformy a jazyky

4.1 EAN

European article number je evropským standartem čárových kódů. Je nadstavbou UPC. Jako UPC jednoznačně identifikuje výrobek a jeho výrobce. Kód může být přečten jak přímým směrem tak i opačným. Používá numerickou symboliku pevné délky. Každý symbol je kódován 2 čáry a 2 mezery. Nejrozšířenější typ je EAN-13. Používá se ke kódování 13-místného čísla. Zpravidla první 2 symboly značí systém číslování a stát, dalších 5 je identifikační číslo výrobce, dalších 5 je kód výrobku a poslední číslice je kontrolní symbol.

Sestavení kódu:
SLLLLLLLMRRRRRRE

- S,M,E jsou ochranné čáry. S a E jsou sestaveny ze 3 unitů ČBČ. M je sestavená z 5 unitů BČBČB.
- L jsou symboly levé poloviny. L čísla jsou vždy sestavené BČBČ. Každý symbol je kódován 7 unity. První symbol(symbol státu) se kóduje způsobem kódování levé strany kódu. Například číslo 1 je kodováno jako LLGLGG, kde G jsou zrcadlové R.
- R jsou symboly pravé poloviny. R symboly jsou zobrazené v negativu – ČBČB. Každý symbol je kódován 7 unity.

Výpočet kontrolního symbolu se provádí podle algoritmu modulo 10:

1. Součet všech čísel na lichých pozicích (1, 3, 5) krát 3.
2. Součet všech čísel na sudých pozicích (2, 4, 6).
3. Výsledky prvních dvou kroku se sčítají a najdeme zůstatek od dělení tohoto součtu 10.
4. Výsledek 3. kroku odečteme od 10. V případě, že výsledek 3.kroku je 0 odečítaní neprovádíme a 0 je kontrolním symbolem.

Rozměrové charakteristiky:

- Tloušťka jednoho unitu $0.33mm$ (měřítko 100%).
- Povolena šířka jedné čáry 1..4 unity.

- Přípustné rozměry 80%-200%.
- Standardní šířka kódu 37.29mm (včetně klidových zón).
- Standardní vertikální délka čáry 22.85mm(výjimkou jsou ochranné čáry).

Podtypy EAN:

- EAN-13 - 13-místný kód. Vlastností jsou popsané výš. Je nejpoužívanějším kódem z rodiny EAN.
- EAN-8 – 8-místný kód. Na rozdíl od UPC-E EAN-8 je samostatný kód. Používá kódování EAN-13. Poslední symbol je kontrolní. Výpočet kontrolního symbolu se provádí podle stejného algoritmu modulo 10. Má levou pravou a střední ochranné čáry EAN-128(GS1-128) – kód proměnné délky. Používá slovník Code128. Používá se pro kódování informací o nákladu mezí společnostmi.
- EAN-2 a EAN-5 – podpůrné kódy. Používají se ve spojení s jiným EAN.

Poznámka: JAN je EAN-13 s kódem státu 45 nebo 49(kód Japonska).

4.2 UPC

Universal Product Code - univerzální kód výrobků. Americký standart čárových kódů. Jednoznačně identifikuje výrobek a jeho výrobce. Kód může být přečten jak přímým směrem tak i opačným. Používá souvislou numerickou symboliku pevné délky. Každý symbol je kódován 4 čáry. Nejrozšířenější typ je UPC-A. Používá se ke kódování 12-místného čísla. Zpravidla první symbol značí systém číslování, dalších 5 je identifikační číslo výrobce, dalších 5 je kód výrobku a poslední číslice je kontrolní symbol.

Sestavení kódu:
SLLLLLLMRRRRRRE

- S,M,E jsou ochranné čáry. S a E jsou sestaveny ze 3 unitu ČBČ. M je sestavená z 5 unitu BČBČB.
- L jsou symboly levé poloviny. L čísla jsou vždy sestavené BČBČ. Každý symbol je kódován 7 unity.
- R jsou symboly pravé poloviny. R symboly jsou zobrazené v negativu – ČBČB. Každý symbol je kódován 7 unity.

Výpočet kontrolního symbolu se provádí podle algoritmu modulo 10:

1. Součet všech čísel na lichých pozicích (1, 3, 5) krát 3.
2. Součet všech čísel na sudých pozicích (2, 4, 6).
3. Výsledky prvních dvou kroku se sčítají a najdeme zůstatek od dělení tohoto součtu 10.
4. Výsledek 3. kroku odečteme od 10. V případě, že výsledek 3.kroku je 0 odečítaní neprovádíme a 0 je kontrolním symbolem.

Rozměrové charakteristiky:

- Tloušťka jednoho unitu 0.33mm (měřítko 100%).
- Povolena šířka jedné čáry 1..4 unity.

- Přípustné rozměry 80%-200%.
- Standardní šířka kódu 37.29mm (včetně klidových zón).
- Standardní vertikální délka čáry 22.85mm(výjimkou jsou ochranné čáry).

Podtypy UPC:

- UPC-A – 12-místný kód. Vlastností jsou popsané výš. Je nejpoužívanějším kódem z rodiny UPC.
- UPC-B – 12-místný kód. Neobsahuje kontrolní symbol. Nabyvá 1 symbol pro kód výrobku.
- UPC-C – 12-místný kód. Obsahuje kód vyrobku a kontrolní symbol.
- UPC-D – kód proměnné délky (12 symbolů a víc), kde 12. symbol je kontrolní.
- UPC-E – 8-místný kód. Ekvivalentní UPC-A. Potlačuje nuly. Má jen levou a pravou ochranné čáry.
- UPC-2 a UPC-5 – podpůrné kódy. Používají se ve spojení s jiným UPC.

4.3 Interleaved 2of5

Samoopravný numerický kód. Ke kódování se používá tenké a tlusté čáry. Každý symbol je kódován 5 čárami, z nichž 2 jsou tlusté a 3 jsou tenké. Čísla jsou kódovány párem (jedno se kóduje čárami, druhé mezerami), zvětšuje se tím hustota kódování. Z toho důvodu obecně je kódován sudy počet čísel, v případě lichého počtu symbolu připisuje se úvodní 0 nebo kontrolní symbol. Kontrolní symbol v kódu být nemusí.

Sestavení kódu: *SPP...PE*

- S je start symbol. Vždy vypadá jako 1010(tenká čára, tenká mezera, tenká čára, tenká mezera).
- E je stop symbol. Vždy vypadá jako 1101(tlustá čára, tenká mezera, tenká čára).
- P jsou páry čísel. Každý pár je sestaven z 5 čar a 5 mezer.

Výpočet kontrolního symbolu se provádí podle algoritmu modulo 10:

1. Součet všech čísel na lichých pozicích (1, 3, 5) krát 3.
2. Součet všech čísel na sudých pozicích (2, 4, 6).
3. Výsledky prvních dvou kroku se sčítají a najdeme zůstatek od dělení tohoto součtu 10.
4. Výsledek 3. kroku odečteme od 10. V případě, že výsledek 3.kroku je 0 odečítaní neprovádíme a 0 je kontrolním symbolem.

Rozměrové charakteristiky:

- Tloušťka jednoho unitů 1.02mm (měřítko 100%).
- Šířka tlusté čáry 2.25-3 unitů.
- Přípustné změny rozměrů 25%-100%.
- Klidová zóna musí být minimálně 10 unitů nebo .25 palců (6.35mm) .
- Výška musí být minimálně 0.15 krát celková šířka kódu. Doporučená výška je 1.25 palců (31.75mm).

Nejrozšířenější implementací Interleaved2of5 je ITF-14. ITF-14 koduje 14 čísel. Poslední symbol musí být kontrolní(počítá se podle algoritmu popsaného výš).

4.4 Code 128

Vysoce efektivní kód proměnné délky, souvislý. Používá alfanumerickou symboliku . Každý symbol je kódován 3 čáry a 3 mezery. Celková šířka jednoho symbolu je 11 unitu. Code 128 má 3 mody kódování a způsob přechodu mezí nimi uvnitř kódu. Každý mód má symboly s kódem od 0 do 105, který současně je jejich vahou. Každý mód má taky možnost překlopení do jiných módu a funkční symboly. Kód používá se zejména pro kódování velkých počtu informací na omezeném prostoru.

Sestavení kódu:
$SAAA...ACE$

- S je start symbol, který taky označí v jakem modu je kódována informace.
- A jsou datové symboly. Váha každého symbolu je od 0 do 105.
- C je kontrolní symbol.
- E je stop symbol.

Výpočet kontrolního symbolu se provádí podle algoritmu modulo 103:

1. Přičtete váhu Start symbolu k výslednému součtu. (Na začátku výsledný součet je 0. Po prvním kroku bude 103, 104 nebo 105).
2. Váhy všech datových symbolu vynásobíte její pozicí a sečtete je a přičtete k výslednému součtu. (První datový symbol má pozice 1).
3. Najděte zůstatek od dělení výsledného součtu 103. Zůstatek od tohoto dělení je kontrolním symbolem.

Rozměrové charakteristiky:

- Tloušťka jednoho unitů $0.33mm$ (měřítko 100%).
- Povolena šířka jedné čáry 1..4 unity.
- Přípustné rozměry 80%-200%.
- Klidová zóna musí být minimálně 10 unitů nebo .25 palců ($6.35mm$)
- Výška musí být minimálně 15% celkové šířky kódu. Minimální výška je .25 palců ($6.35mm$).

4.5 Code 11

Code 11, taky známy jako USD-8, je numerický kód proměnné délky s vysokou hustotou kódování. Každý symbol je sestaven z 3 čar a 2 mezer a jsou odděleny od sebe tenkou mezerou. Pro kódování používá tenké a tlusté čáry a mezery. Každý symbol je sestaven z tlusté čáry, jednoho tlustého elementu (čáry nebo mezery) a tenkých elementů. Abeceda kódu je 11 symbolů (0-9, -, $*$). Používá se zejména v telekomunikacích.

Sestavení kódu:
SAA...ACKE

- S je start symbol. Start symbolem je hvězda.
- A jsou datové symboly.
- C je kontrolní symbol.
- K je kontrolní symbol. Používá se, když kód má víc než 10 symbolů.
- E je stop symbol. Stop symbol je taky hvězda.

Výpočet kontrolního symbolu C se provádí podle algoritmu modulo 11:

1. Sečtete váhy všech datových symbolu (start a stop symboly nejsou datové, „-“ má váhu 10) vynásobené jejích pozicí(počítání pozicí začíná se od práva číslem 1, maximální pozice je číslo pozice je 10, po dosažení 10 číslování pozici znovu začíná se od 1).
2. Zůstatek od dělení tohoto součtu 11 je kontrolní symbol C.

Výpočet kontrolního symbolu K se provádí podle algoritmu modulo 9:

1. Sečtete váhy všech datových symbolu a kontrolního symbolu C(start a stop symboly nejsou datové) vynásobené jejích pozicí(počítání pozicí začíná se od práva číslem 1, maximální pozice je číslo pozice je 9, po dosažení 9 číslování pozici znovu začíná se od 1. První pozicí tedy má kontrolní symbol C).
2. Zůstatek od dělení tohoto součtu 9 je kontrolní symbol K.

Rozměrové charakteristiky(Tyto charakteristiky jsou předpoklad na základě shodných typu kódů):

- Minimální tloušťka jednoho unitu je $0.19mm$.
- Šířka tlusté čáry 2-3 unitů.
- Klidová zóna musí být minimálně 10 unitu nebo .25 palců ($6.35mm$)
- Výška musí být minimálně 0.15 krát celková šířka kódu. Doporučená výška je .25 palců ($6.35mm$)

4.6 Code39

Code39 nebo Code 3 of 9 je samoopravný alfanumerický kód proměnné délky. Každý symbol je sestaven z 5 čar a 4 mezer. Z 9 prvku 3 jsou široké, odkud pochází název kódu. Code 39 dovoluje zakódovat velká písmena (A-Z), číslice (0-9) a několik symbolu(+, -, /, atd.). Ke kódu muže být připsán kontrolní symbol, avšak není nutný protože kód je samoopravný.

Sestavení kódu: *SAA...ACE*

- S je start symbol. Start symbolem je hvězda.
- A jsou datové symboly.
- C je kontrolní symbol (nemusí být v kódu).
- E je stop symbol. Stop symbol je taky hvězda.

Výpočet kontrolního symbolu se provádí podle algoritmu modulo 43:

1. Sečtete váhy všech datových symbolu (start a stop symboly nejsou datové).
2. Zůstatek od dělení tohoto součtu 43 je kontrolní symbol.

Rozměrové charakteristiky:

- Minimální tloušťka jednoho unitu je $0.19mm$.
- Šířka tlusté čáry 2-3 unitů.
- Klidová zóna musí být minimálně 10 unitu nebo .25 palců ($6.35mm$)
- Výška musí být minimálně 0.15 krát celková šířka kódu. Doporučená výška je .25 palců ($6.35mm$)
- Šířka mezery minimálně je 5.3 unitů když šířka unitu je menší než $0.254mm$. V případě když šířka unitu je větší než $0.254mm$ šířka mezery musí být 3 unity, avšak minimálně $0.135mm$.

Existuje Code39 extended, který umožňuje kódovat 128 symbolu ASCII. Pro kódování rozšířujících symbolu se používají předpony $, /, %, a +. Například „d“ je kódováno jako „+D“ a „—“ je kódováno jako „%Q“.

4.7 Code93

Code 93 je alfanumerický kód proměnné délky. Každý symbol je sestaven z 9 modulu z nichž složené 3 čáry a 3 mezery odkud pochází název kódu. Byl vyvíjen jako vylepšení Code 39, proto kóduje 43 symbolů Code 39 a 4 symboly navíc. Na rozdíl od Code 39 má vyšší hustotu kódování a 2 kontrolní symboly. Code 93 dovoluje zakódovat velká písmena (A-Z), číslice (0-9) a několik symbolu(+, -, /, atd.). Ke kódu muže být připsán kontrolní symbol, avšak není nutný protože kód je samoopravný.

Sestavení kódu:
SAA...ACKE

- S je start symbol. Start symbolem je hvězda.
- A jsou datové symboly.
- C,K jsou kontrolní symboly.
- E je stop symbol. Stop symbol je taky hvězda.

Výpočet kontrolního symbolu C se provádí podle algoritmu modulo 47:

1. Sečtete váhy všech datových symbolu (start a stop symboly nejsou datové) vynásobené jejích pozicí(počítání pozicí začíná se od práva číslem 1, maximální pozice je číslo pozice je 20, po dosažení 20 číslování pozici znovu začíná se od 1).
2. Zůstatek od dělení tohoto součtu 47 je kontrolní symbol C.

Výpočet kontrolního symbolu K se provádí podle algoritmu modulo 47:

1. Sečtete váhy všech datových symbolu a kontrolního symbolu C(start a stop symboly nejsou datové) vynásobené jejích pozicí(počítání pozicí začíná se od práva číslem 1, maximální pozice je číslo pozice je 15, po dosažení 15 číslování pozici znovu začíná se od 1. První pozicí tedy má kontrolní symbol C).
2. Zůstatek od dělení tohoto součtu 47 je kontrolní symbol K.

Rozměrové charakteristiky:

- Minimální tloušťka jednoho unitu je $0.19mm$.
- Šířka čáry může být 1..4 unity.
- Klidová zóna musí být minimálně 10 unitu nebo .25 palců ($6.35mm$)
- Výška musí být minimálně 0.15 krát celková šířka kódu. Doporučená výška je .25 palců ($6.35mm$)

Existuje Code93 extended, který umožňuje kódovat 128 symbolu ASCII. Pro kódování rozšiřujících symbolu se používají ty 4 speciální symboly, o které Code 93 rozšiřuje Code 39, jako předpony.

4.8 Codabar

Codabar je diskrétní samoopravný kód. Slovník kódu obsahuje 16 symbolů (0-9 a $, :, /, ., +, -). Navíc kód používá 4 start/stop symboly (A,B,C,D). Každý symbol je sestaven z 4 čar a 3 mezer, každý symbol obsahuje 1 tlustou čáru a 1 tlustou mezeru. Výjimkou jsou symboly :,/,+,., které obsahují 3 tlusté čáry a všechny mezery jsou tenké.

Sestavení kódu:
SAA..AE

- S,E jsou start/stop symboly.
- A jsou datové symboly.

Rozměrové charakteristiky:

- Minimální tloušťka tenké čáry je $0.165mm$.
- Povolena šířka tlusté čáry od 2.25 do 3 unity.

4.9 MSIPlessey

MSI plessey, taky známý jako modifikovány Plessey, je numerický kód libovolné délky. Každý symbol je kódován 4 moduly (modul je složení čáry a mezery). Existuje 2 moduly, 1 – tlustá čára tenká mezera a 0 – tenká čára tlustá mezera. Kód může obsahovat 1 nebo 2 kontrolní symboly. Kontrolní symboly můžou být v těchto variacích: Modulo 10, 2x Modulo 10, Modulo 11, Modulo 11 + Modulo 10.

Sestavení kódu:
SAA..ACKE

- S,E jsou start/stop symboly (start symbolem je tlustá čára a tenká mezera, stop symbolem je tenká čára tlustá mezera a tenká čára).
- A jsou datové symboly.
- C je kontrolní symbol.
- K je kontrolní symbol. Nemusí být v kódu.

Výpočet kontrolního symbolu se provádí podle algoritmu modulo 10. Pozor! Je jiný, než předchozí:

1. Sestavte nové číslo z každého druhého symbolu kódu. Vynásobíte toto číslo 2.
2. Sečtete všechny symboly které zůstali v čárovém kódu (tedy každý na liché pozice počínaje zleva). K výsledku přečtěte všechny číslice výsledku kroku 1(například výsledek kroku 1 je 123, pak přečteme 1+2+3).
3. Najděte zůstatek od dělení tohoto součtu 10.
4. Výsledek 3. kroku odečteme od 10. Tohle číslo je kontrolním symbolem.

Výpočet kontrolního symbolu se provádí podle algoritmu modulo 11:

1. Vynásobte každý symbol jeho vahou (váha počínaje zprava číslem 2. S každým krokem se zvětšuje o 1. Maximální váha je 7. Po dosažení maximální váhy následující váha je 2, 3 atd.)
2. Najděte zůstatek od dělení tohoto součtu 11.
3. Výsledek 3. kroku odečteme od 11. Tohle číslo je kontrolním symbolem.

V případě kontrolních symbolů 2x modulo 10 provádí se výpočet prvního kontrolního symbolů, pak se provádí výpočet druhého s podmínkou, že první kontrolní symbol je zapsán do kódu.

V případě kontrolních symbolů modulo 11 + modulo 10 provádí se výpočet prvního kontrolního symbolů (Modulo 11), pak se provádí výpočet druhého (Modulo 10) s podmínkou, že první kontrolní symbol je zapsán do kódu.

Rozměrové charakteristiky:

- Standardní tloušťka modulu(černá čára a bíla mezera) je 1.02, přičemž pro modul 1 šířka čáry musí ležet v rozmezí od $0.533mm$ do $0.635mm$ a šířka mezery v rozmezí od $0.381mm$ do $0.483mm$. Pro modul 0 šířka čáry musí ležet v rozmezí od $0.127mm$ do $0.229mm$ a šířka mezery od $0.787mm$ do $0.889mm$.

4.10 PostNet

PostNet byl vyvíjen pro poštovní systém USA a muže kódovat Zip kódy(tvoří 5 čísel a je analogem PSČ v USA) , Zip+4 kódy a plný poštovní kód. Kód na rozdíl od ostatních kóduje informací ne pomocí šířky čáry ale pomocí její délky. PostNet vždycky obsahuje kontrolní symbol.

Sestavení kódu:
SAA..ACE

- S,E jsou start/stop čáry. Jedna vysoká čára.
- A jsou datové symboly.
- C je kontrolní symbol.

Výpočet kontrolního symbolu se provádí podle algoritmu modulo 10. Pozor! Je jiný než předchozí:

1. Sečtěte všechny symboly.
2. Najděte zůstatek od dělení tohoto součtu 10.

3. Výsledek 3. kroku odečteme od 10. Výsledné číslo je kontrolním symbolem. V případě že výsledek 3. kroku je 0, kontrolní symbol je 0.

Rozměrové charakteristiky:

- Doporučená šířka čáry je od $0,381mm$ do $0,635mm$.
- Doporučená šířka mezery od $0,3048mm$ do $1,016mm$.
- Doporučená výška malé čáry $1,4478mm$.
- Doporučená výška velké čáry $2,8194mm$.

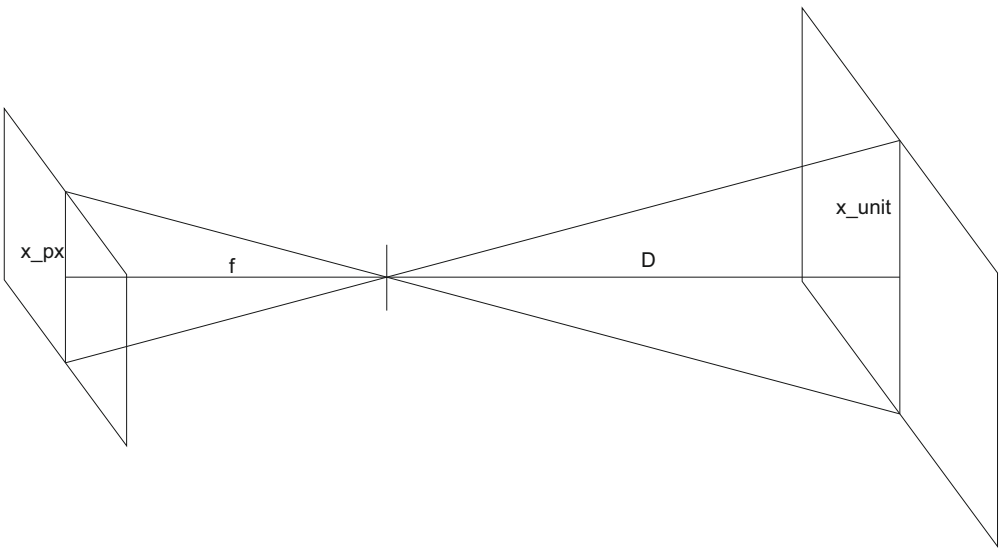
Kapitola 5
Příprava experimentů

Experimenty byly vykonány v laboratoři robotické vize. K provedení experimentů byly ploužíte tyto přístroje:

- Kamera: F8825A0 Q-TECH
- Držák pro kódy
- Držák pro kameru
- Kolejnice: 180cm
- Příslušenství k osvětlení: 2x žárovka 100W
- Ruleta
- Expozimetr: Sekonic L-508 zoom master

5.1 Parametry kamery

Pro vykonání experimentů byla použita kamera F8825A0 Q-TECH vestavěná do telefonu Lenovo p780 v režimu 8.0M (3264x2448 pixelů) . Počet aktivních pixelů matici je 3600x2448 pixelů. Úhel pohledu kamery a vzdálenost mezi čočkou a maticí byly stanoveny experimentálně, kvůli nedostupnosti technického manuálu pro tuto kameru. Rozměry kamery jsou 8.5mm x 8.5mm (změřené ručně), jako analogii používám kameru 08413MA/QX214AR nebo 09613MA/Q13850B (matici těchto kamer mají velmi shodné parametry). Uvažujeme zjednodušený model kamery.



Obrázek 5.1. Zjednodušený model kamery.

Pro zjištění úhlu pohledu kamery a taky distance mezi matici a čočkou vyfotíme objekt známých rozměrů. Jako objekt známých rozměru používám papír A4, jeho rozměry jsou 297mm x 210mm.

Nastavíme vzdálenost mobilu od papíru tak, aby kratší hranice snímku se splývali s kratšími stranou papíru a změříme vzdálenost.
Změřená vzdálenost je 242mm.
Podle formule:

$$\alpha_H = arctg\left(\frac{x_{A4}}{2D_{Cam}}\right)$$

kde
α_H - horizontální úhel pohledu.
x_{A4} - délka papíru A4 (297mm).
D_{Cam} - vzdálenost kamery od objekta.
Po dosazení dostaneme $\alpha_H = 31.53^o$

Nastavíme vzdálenost mobilu od papíru tak, aby delší hranice snímku se splývali s kratšími stranou papíru a změříme vzdálenost.
Změřená vzdálenost je 325mm.
Podle formule:

$$\alpha_V = arctg\left(\frac{y_{A4}}{2D_{Cam}}\right)$$

kde
α_V - vertikální úhel pohledu.
y_{A4} - šířka papíru A4 (210mm).
D_{Cam} - vzdálenost kamery od objekta.
Po dosazení dostaneme $\alpha_H = 24.56^o$

Vypočítáme distance mezí čočkou a maticí, za předpokladu, že rozměry matici 4815μm x 3678.3μm, známého maximálního možného rozlišení kamery v horizontální ploše a známého poměru použitých pixelů v režimu 8.0M k maximálnímu počtu pixelů.
Podle formule:

$$f = \frac{x_H \frac{N_{HA}}{N_{HV}}}{2tan(\alpha_H)}$$

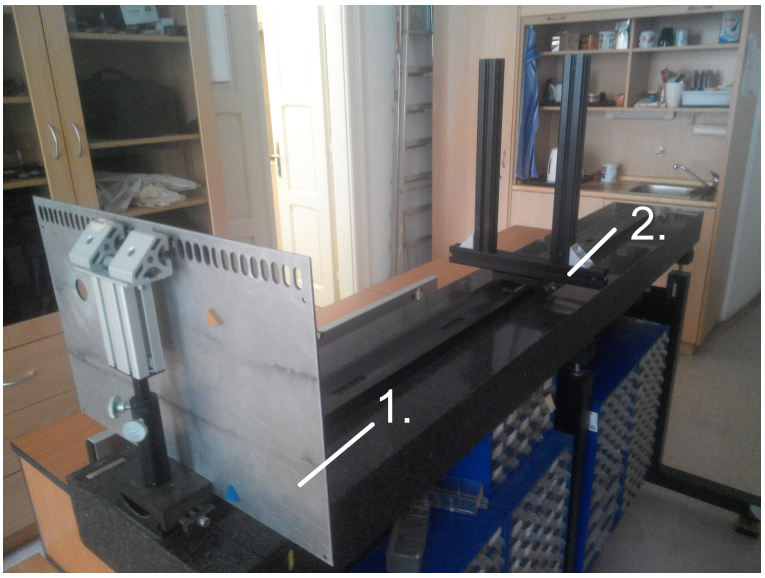
kde x_H - šířka aktivní části matici kamery.
N_{HA} - počet aktivních pixelů v režimu 8.0M.
N_{HV} - maximální počet aktivních pixelů.
α_H - horizontální úhel pohledu.
f - vzdálenost matici od čočky.
Po dosazení dostaneme $f = 3557.8\mu m$

Parametry kamery (režim 8.0M):

- Rozměry kamery: 8.5mm x 8.5mm x 5.3mm
- Rozměr aktivní části matici 4365.6μm x 3678.3μm
- Rozlišení: 3264 x 2448
- Vzdálenost matici od čočky: 3557.8μm
- Vertikální úhel pohledu: 31.53^o
- Horizontální úhel pohledu: 24.56^o

Obrázek 5.2. Vnější pohled na kameru F8825A0 Q-TECH.

5.2 Sestavá pro provedení experimentů



Obrázek 5.3. Sestavá pro provedení experimentů.

kde

1. je držák pro kódy. Kódy jsou upevněné pomocí magnetů.
2. je držák pro kameru.

5.3 Software pro experimenty

5.3.1 Generátor čárových kódů

Pro testy zvolíme maximálně rozšířený čárový kód a to je EAN-13. Ke generací čárových kódů byl použit generátor Zint[8]. Pro experimenty byly vygenerovány tyto EAN-13 kódy:

- 0623245000019
- 0518437571036

- 2412345678901
- 3452910548726
- 3803949352684
- 4453217584942
- 5903242343438
- 7350053850033
- 8260635277921
- 9312345678907

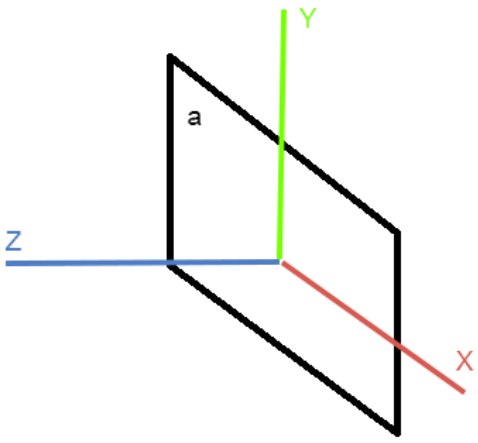
Všechny kódy byly vygenerovány v měřítku 100%, rozmístěny každý na svou stránku A4 a vytištěné pomocí tiskárny s kvalitou tisku 600dpi. (Materiály pro výtisk testovacích vzorů jsou v příloze).

5.3.2 Scripty pro zpracování obrázků

Skripty pro otáčení obrázků používají tento algoritmus:

1. Vytvořit model reálného objektu v pamětí na základě obrázku. (reálným objektem je myšlená scéna kterou zachytil obrázek).
2. Otočíme reálný objekt.
3. Simulujeme projekcí nového objektu na matici kamery.

Při otáčení obrázků používám tuto osovou konvenci.



Obrázek 5.4. Osova konvence.

Zavedeme lokální terminologickou konvenci:

- starý pixel – pixel z originálního obrázku, který musí být přemapován.
- nový pixel – pixel z vytvořeného pomocí scriptu obrázku.

Protože při otáčení objektu okolo os X a Y se vytvářejí oblasti se zvýšenou hustotou pixelů a sníženou hustotou pixelů, pro tyto oblastí použijeme různé algoritmy zpracování. Pro oblasti zvýšené hustoty nový pixel převezme barvu nejbližšího starého pixelů. Při otáčení okolo osy X se vytvářejí horizontální oblasti se sníženou hustotou pixelů. Nevyplněné řádky převezmou barvu vypočtenou jako vážený součet barev nejbližších řádků. Při otáčení okolo osy Y se vytvářejí vertikální oblasti se sníženou hustotou pixelů.

Nevyplněné sloupce převezmou barvu vypočtenou jako vážený součet barev nejbližších sloupců.

Při otáčení objektu okolo osy Z nejdřív vyplníme nové pixely barvou nejbližších starých pixelů. Nové pixely, které nejsou vyplněné, vyplníme na základě barev okolních vyplněných pixelů.

Taky používá se script pro oříznutí obrázků, protože Manatee, Scandit a Zxing očekávají omezení oblasti ve které mají hledat čárový kód.

5.4 SDK a realizace jejich použití

Nejpohodlnější platformou pro testování možnosti SDK bude PC. Všechny SDK jsou integrovatelné do velkého množství platforem a všechny jsou integrovatelné do Android a iOS, což jsou cílové platformy, ale při integrací do Android je velmi obtížně (někdy nemožné) SDK donutit skenovat kód z obrázek, ne z videostreamu. Videostreamy se používají většinou v režimu 720p(Scandit ve testovací aplikaci má možnost nastavit režim 1080p), což je 2.5 krát menší rozlišení, než při zachytnuti obrázku. Proto používáme testovací SDK pro Linux a Java.

5.4.1 ZXing

SDK ZXing je stále ve vývoji, základním jazykem je Java. Cílovou platformou je Android a JVM, ale existují porty do C++, iOS, Objective C, Jruby. Porty do jiných jazyků nejsou udržované (poslední commit v C++ portu byl 9. září 2013[17]). Proto budeme používat Java distribuci. ZXing je složitě integrovatelný, a především jeho cílem je přečtení čárových kódů z videostreamu. Při pokoušení do základního SDK nahrát čárový kód, vždy nemohl ho najít na obrázku, což je způsobeno tím, že potřebuje silně omezenou oblast ve které hledá kód. Abych ho mohl otestovat použil jsem QRDroid[13], což je aplikace pro čtení čárových kódů od Zapper, která používá ZXing knihovnu. Aplikaci se dá předat obrázek na zpracování a dostat výsledek. Děla se to pomocí „intent“ na platformě Android. Napsal jsem jednoduchou aplikaci pro použití QRDroid. Pro práci potřebuje instalovanou QRDroid aplikaci.

5.4.2 ZBar

SDK se vyvíjelo v C. K provedení experimentů použijeme zbarimg, což je program pro načítání čárových kódů z obrázku. Dá se ho instalovat jak z poskytnutých na webových stránkách SDK zdrojových souboru, tak i stáhnout jako zkompilovaný program pro Ubuntu z „universal“ repositáře. K práci potřebuje nainstalovaný ImageMagick ve verzi minimálně 6.2.6 a pkg-config. Po instalací zbarimg jednoduše používáme ho přes shell script. Aplikace vypisuje informací o času zpracování na chybový vystup, proto potřebujeme jeho přesměrovat.

5.4.3 Scandit SDK

Poskytuje možnost integrací do různých platforem. Pro Linux SDK má předkompilovanou testovací aplikací pro načítání kódů z obrázků. Tuto aplikaci budeme používat k testování SDK. Aplikace vyvoláme pomocí shell scriptu a jimž změříme strávený aplikací čas na přečtení kódů, protože aplikace neposkytuje informací o času zpracování. Aplikace snaží se při každém vyvolání se připojit na internet, což pravděpodobně zvyšuje čas stráveny na přečtení kódu. K načítání obrázků používá ImageMagick. Testovací licence se poskytuje na dobu 30 dnů pro 20 zařízení, každým můžete naskenovat 500 kódů.

5.4.4 ManateeWorks SDK

SDK může být integrované do různých platforem. Linuxová distribuce poskytuje zkompilovanou testovací aplikaci, která dokáže číst čárové kódy z obrázků ve formátu .jpg. Aplikace nevyžaduje žádné pomocné programy a poskytuje informací o času načítání kódu. Aplikace vyvoláme pomocí shell scriptu. Licence se poskytuje na 30 dnů.

5.4.5 Jiné

i-Nigma: neposkytuje testovací licencí.
VSBarcodeScanner: neposkytuje testovací licencí.

5.5 Postup záchytu obrázků.

Cílem úlohy je otestovat SDK. Pro lepší kvalitu obrázků, pokusíme se udělat co nejlepší osvětlení kódů, čímž snížíme úroveň výsledného zašumění obrázků. Postup při provedení záchytu obrázků:

1. Upevníme čárový kód na držáku pro čárové kódy tak, aby centr čárového kódů, což je střední ochranná čára byla ve středu kamery.
2. Nastavíme osvětlení tak, aby úroveň odraženého světla byl 640 lx.
3. Nastavíme držák čárových kódů do požadováné vzdálenosti od kamery a uděláme snímek
4. Zopakujeme kroky 2. a 3. pro všechny potřebné vzdálenosti.
5. Zopakujeme kroky 1.-4. pro všechny čárové kódy.

Kapitola 6
Experimenty

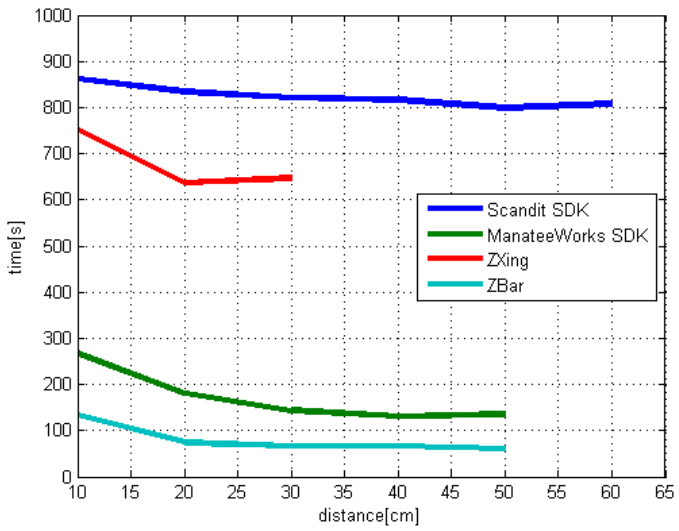
6.1 Maximální distance a čas zpracování kódů

Provedeme test vzdálenosti a času, které SDK potřebují pro přečtení čárových kódů. Předpokladem je, že se zvětšením vzdálenosti se bude zvětšovat čas zpracování obrázku za předpokladu, že horší rozlišení obrázku způsobí delší práci předzpracování obrázku. Použijeme napsané pro SDK scripty na připravenou sadu testovacích obrázků. Obrázky byly ořízнuté, protože některé SDK potřebují vyhrazenou oblast pro hledání kódů.

distance[cm]	Scandit[ms]	Manatee[ms]	ZXing[ms]	ZBar[ms]
10	862.4	268.2	751.0	134.0
20	833.6	180.1	636.5	74.0
30	821.2	143.6	646.5	67.0*(9)
40	816.0	130.2	x	67.0*(9)
50	799.2	135.6	x	60.0
60	807.6	x*(7)	x	x
70	x	x	x	x

Tabulka 6.1. Čas zpracování čárového kódů v zavilosti na distancí.

Pro přehlednost vykreslíme data v podobě grafu.



Obrázek 6.1. Čas zpracování čárového kódů v zavilosti na distancí.

Scandit SDK ukázalo největší vzdálenost načítání a dokázalo přečíst 8/10 čárových kódů ze vzdálenosti 60cm. Časově se projevilo jako nejhorší, předpokládám, že je to

spojené s tím, že SDK po každém vyvolání se pokouší připojit ke licenčnímu serveru. SDK potřebuje vyhrazenou oblast, ve které bude hledat čárový kód.

ManateeWorks SDK, používá větší čas na zpracování kódů než ZBar, ale pravděpodobně je to způsobené kontrolou licencí, kterou provádí po každém vyvolání. Maximální vzdálenost načítání kódů za těchto podmínek je 50cm. Nedokázalo přečíst žádný kód ze distanci 60cm ani zpracované obrázky ve distance 55cm. Pro jeden kód ze distance 60cm vypsalo chybnou odpověď. SDK potřebuje vyhrazenou oblast, ve které bude hledat čárový kód.

ZXing má nejmenší vzdálenost načítání (maximálně 30cm), což nesplňuje ani předpoklad vyplývající z Shannonůva teorému, tedy že libovolné SDK by mělo přečíst kód ze vzdálenosti 43.89cm. Čas ztracený SDK na zpracování rychlé klesá s plochou zábranou na obrázku kódem. SDK potřebuje velmi omezenou oblast, ve které bude hledat čárový kód.

ZBar je nejrychlejší SDK, maximální vzdálenost spolehlivého načítání kódů za těchto podmínek má 50cm. Nedokázalo přečíst žádný kód ze vzdálenosti 60cm. Dokázalo ale přečíst 3 ze 10 kódů zpracovaných tak aby odpovídali obrázkům udělaným ze distance 55cm. SDK dokáže vyhledat čárový kód na obrázku s velkou plochou, na rozdíl od jiných SDK, které potřebují vyhrazenou oblast, ve které budou hledat čárový kód.

Ukázala se taky tendenci ke klesání času ztráceného na zpracování obrázků s růstem vzdálenosti(předpoklad byl obrácený). Malý růst času zpracování je vidět na hranici maximální distanci. Z toho můžeme udělat mezilehlý závěr, že předzpracování zabírá víc času, ale až na hranici možností SDK, což je způsobeno rychlým poklesem rozlišení pří větších vzdálenostech.

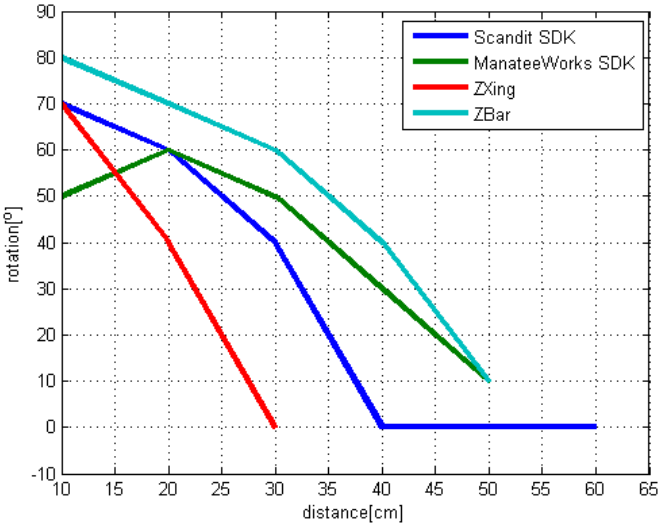
6.2 Vliv rotace okolo osy Y na vzdálenost načítání kódů

Provedeme ocenění vlivu rotace okolo osy Y na vzdálenosti načítání čárových kódů. Cílem je najít meze načítání čárových kódů pomocí SDK za podmínky zmenšeného počtů pixelů na unit. Zjistíme maximální úhel natočení čárového kódu pro každou vzdálenost při kterém se kód stále dá spolehlivě přečíst.

vzdálenost[cm]	Scandit[^{o}]	Manatee[^{o}]	ZXing[^{o}]	ZBar[^{o}]
10	70	50	70	80
20	60	60	40	70
30	40	50	0	60
40	0	30	x	40
50	0	10	x	10
60	0	x	x	x

Tabulka 6.2. Maximální přípustná rotace obrázku okolo osy Y při načítání z různých vzdálenosti.

Pro přehlednost si tato data zobrazíme v podobě grafu.



Obrázek 6.2. Maximální přípustná rotace obrázku okolo osy Y při načítání z různých vzdálenosti.

Z grafu je vidět, že nejlepším SDK pro načítání čárových kódů za podmínky rotace okolo osy Y je ZBar. Dál jde ManateeWorks SDK, které potřebovalo menší úhel rotací na minimální vzdálenosti, což je pravděpodobně způsobeno tím, že SDK ne uvidělo polohu kódu na obrázku, tedy nesnažilo se ho přečíst. Třetí je Scandit SDK které ukázalo rychlý pokles možného natáčení kódů s růstem vzdálenosti.

6.3 Vliv sníženi kontrastu na SDK

Zavedeme si lokální jednotku kj(koeficient sníženi kontrastu), která bude odpovídat sníženi kontrastu pomocí matlabové funkce imadjust(I,[low_in; high_in],[low_out; high_out]). Zvýšením low_out na 0.05 a snížením high_out o 0.05 dokud low_out a high_out nebudou si rovné, dostaneme sadu obrázku se sníženým kontrastem. Jednotka kj – je násobek 0.05k(kde k je celé číslo) vynásobený 100. Střední hodnota kontrastu u zpracovaných obrázku v závislosti na vzdálenosti a kj je znázorněna v tabulce niž 6.4.

Najdeme nejnižší možný kontrast, při kterém kód se bude spolehlivě načítat, pro každou vzdálenost v rozmezí 10..60 (tedy měřené vzdálenosti načítání kódů).

distance[cm]	Scandit[kj]	Manatee[kj]	ZXing[kj]	ZBar[kj]
10	35	45	40	45
20	40	45	40	45
30	40	45	35	45
40	40	45	x	45
50	40	45	x	40
60	40	x	x	x

Tabulka 6.3. Potřebný kontrast pro úspěšné načítání kódu v závislosti na vzdálenosti obrázku.

Z výsledků je vidět, že SDK dokážou pracovat s velmi nízkými kontrasty. Manatee ukázalo nejlepší výsledek, že výborně pracuje při nejnižším kontrastu a neovlivnil kontrast vzdálenost načítání. Scandit ukázalo, že kontrast neovlivni vzdálenost načítání,

kj	EX	σ	kj	EX	σ
dist. 10			dist. 40		
0	0.8865	0.0138	0	0.8590	0.0303
5	0.8541	0.0167	5	0.8269	0.0279
10	0.8135	0.0192	10	0.7852	0.0260
15	0.7617	0.0220	15	0.7421	0.0300
20	0.7217	0.0255	20	0.6962	0.0284
25	0.6332	0.0168	25	0.6151	0.0159
30	0.4713	0.0098	30	0.5098	0.0142
35	0.3930	0.0098	35	0.4303	0.0107
40	0.2803	0.0101	40	0.3240	0.0142
45	0.1627	0.0057	45	0.1869	0.0070
dist. 20			dist. 50		
0	0.8538	0.0111	0	0.8856	0.0172
5	0.8195	0.0085	5	0.8510	0.0177
10	0.7810	0.0100	10	0.8076	0.0188
15	0.7294	0.0079	15	0.7587	0.0149
20	0.6873	0.0087	20	0.7048	0.0160
25	0.6207	0.0083	25	0.6365	0.0162
30	0.4844	0.0082	30	0.5111	0.0087
35	0.4097	0.0100	35	0.4246	0.0082
40	0.2977	0.0083	40	0.3149	0.0104
45	0.1826	0.0080	45	0.1912	0.0077
dist. 30			dist. 60		
0	0.9084	0.0184	0	0.9292	0.0196
5	0.8706	0.0148	5	0.8874	0.0211
10	0.8223	0.0143	10	0.8409	0.0240
15	0.7717	0.0206	15	0.7826	0.0213
20	0.7180	0.0211	20	0.7265	0.0242
25	0.6491	0.0231	25	0.6454	0.0284
30	0.5224	0.0105	30	0.5255	0.0140
35	0.4376	0.0100	35	0.4289	0.0132
40	0.3214	0.0073	40	0.3091	0.0107
45	0.1918	0.0053	45	0.1796	0.0074

Tabulka 6.4. Střední hodnota a směrodatná odchylka kontrastu testované sady obrázků v závislosti na vzdálenosti a použitém koeficientu sníženi kontrastu.

potřebuje ale větší kontrast než Manatee. Ostátní SDK na hraniční vzdálenosti načítání potřebovali větší kontrast. Scandit potřebovalo větší kontrast na menší vzdálenosti, což je pravděpodobně způsobeno tím, že nenašlo čárový kód na obrázku.

Kontrast obrázku se počítal podle formule:

$$k = \frac{B_{max} - B_{min}}{B_{max}}$$

kde
K - kontrast obrázku.
B_{max}, B_{min} - největší a nejmenší úroveň jasu.

6.4 Stanovení rozlišení potřebného pro úspěšné přečtení kódů.

Z výsledků z tabulky 6.1 a z tabulky 6.2 můžeme stanovit rozlišení(počet pixelů na unit), které SDK potřebují pro úspěšné přečtení kódů. Z parametrů kamery:

- Rozměry kamery: 8.5mm x 8.5mm x 5.3mm
- Rozměr aktivní části matici 4365.6μm x 3678.3μm
- Rozlišení: 3264 x 2448
- Vzdálenost matici od čočky: 3557.8μm
- Vertikální úhel pohledu: 31.53^o
- Horizontální úhel pohledu: 24.56^o

Zjistíme, že rozměr jednoho pixelu na maticí je 1.3375μm. Pak podle zjednodušeného modelu kamery2.1 potřebné rozlišení vypočteme jako

$$\frac{k * x_{px}}{f} = \frac{x_{unitu}}{D}$$

$$k = \frac{f x_{unitu}}{D x_{px}}$$

kde
k - počet pixelů na unit.
x_{px} - rozměry(šířka) jednoho pixelu.
f - vzdálenost čočky od matici.
x_{unit} - šířka jednoho unitu.
D - vzdálenost obrázku od kamery.

Po dosázení maximálních vzdálenosti načítání kódů:

SDK	Vzdálenost[cm]
Scandit	60
Manatee	50.77
ZBar	50.77
ZXing	30

Tabulka 6.5. Maximální vzdálenost přečtení pro SDK.

50.77cm plyne z výsledků tabulky 6.2 a počítá se jako $\frac{D}{cos\alpha}$

do teto formule dostaneme, že počet pixelů na unit potřebný pro přečtení čárového kódů je:

SDK	Počet pixelů
Scandit	1.38
Manatee	1.63
ZBar	1.63
ZXing	2.76

Tabulka 6.6. Počet pixelů na unit, potřebný pro SDK pro úspěšné přečtení kódu.

Kapitola 7
Závěr

Scandit SDK má nejmenší rozlišení potřebné pro úspěšné přečtení čárového kódu a největší vzdálenost ze které kód může být přečten. Vyžaduje ale větší kontrast než ostatní SDK

Parametr	Výsledek
Potřebné rozlišení unitu	1.38
Maximální vzdálenost načítání kódů	60cm
Střední potřebný kontrast	0.3079

Tabulka 7.1. Parametry Scandit SDK.

ZXing se ukázalo jako SDK s nejvyšším úrovním rozlišení potřebným pro načítání kódů a tedy i s nejmenší vzdáleností, ze které může kód přečíst. Je stále ve vývoji a je obtížně integrovatelné, ale má velkou bázi projektu, které jsou na ZXing založeny.

Parametr	Výsledek
Potřebné rozlišení unitu	2.76
Maximální vzdálenost načítání kódů	30cm
Střední potřebný kontrast	0.2890

Tabulka 7.2. Parametry SDK ZXing.

ManateeWorks SDK je střední ve smyslu maximální vzdáleností načítání a střední v časové efektivnosti. Ukázalo však nejlepší výsledky při práci s různýma kontrasty. ManateeWorks se udržuje a stále se rozvije, má velké množství podporovaných platforem a čárových kódů.

Parametr	Výsledek
Potřebné rozlišení unitu	1.63
Maximální vzdálenost načítání kódů	50.77cm
Střední potřebný kontrast	0.1830

Tabulka 7.3. Parametry ManateeWorks SDK.

ZBar má největší časovou efektivitu a střední vzdálenost načítání. Ukázalo nejlepší výsledek při práci s úhly natočení obrázků a dobrý výsledek při práci s kontrasty. Minusem je, že projekt se již neudržuje a vývoj je zavřen.

Parametr	Výsledek
Potřebné rozlišení unitu	1.63
Maximální vzdálenost načítání kódů	50.77cm
Střední potřebný kontrast	0.1830

Tabulka 7.4. Parametry SDK ZBar.

7.1 Doporučení k použiti

Z výsledku, které ukázali SDK bych řekl, že pro obecné případy užiti doporučil bych Scandit SDK. Má největší vzdálenost přečtení čárového kódu, ale nepracuje moc dobře s otáčeným obrázkem. Počítám tedy, že větší distance přečtení čárového kódu má větší váhu, než možnost otáčení obrázků. Ve práci s obrázky s malým kontrastem projevilo se horší než ostatní SDK, ale rozdíl výsledků nebyl velký a taky jako ostatní SDK dokázalo číst kódy se silně sníženým kontrastem.

Pro případy užiti, kdy je potřebné číst čárové kódy z nepříjemných úhlu pohledu na kód bych doporučil ManateeWorks SDK před Scandit SDK a ZBar. ZBar ukázal stejné výsledky jako Manatee, v testu načítání kódů s rotaci okolo osy Y se projevil líp, ale Manatee na druhou stranu podporuje mnohém víc standardů čárových kódů, je integrovatelné do většího počtů platforem a stále se vyvíjí, což podle mě je hlavní důvod použit Manatee před ZBar.

Osobně bych ohodnotil výkon a použitelnost SDK v tomto pořadí:

1. Scandit SDK
2. ManateeWorks SDK
3. ZBar
4. ZXing

Literatura

[1] spol. s r.o. GABEN. *Gaben*. 2016.
http://www.gaben.cz/cz/faq/carove-kody-teorie.

[2] Adams Communications. *BarCode1*. 1995.
http://www.adams1.com/new.html.

[3] Developer Express Inc. *DevExpress*. 1998-2017.
https://documentation.devexpress.com/#XtraReports/CustomDocument2613.

[4] Inc. BarCodeIsland.com. *Barcode Island*. 2006.
http://www.barcodeisland.com/.

[5] Inc. Bar Code Graphics. *GTIN INFO*. 2017.
http://www.gtin.info/.

[6] IDAutomation.com. *IDAUTOMATION.COM*. 2017.
http://www.idautomation.com/barcode-faq/.

[7] GS1 AISBL. *GS1*. 2017.
http://www.gs1.org/how-calculate-check-digit-manually.

[8] Robin Stuart. *Zint*. 2008 - 2017.
http://www.zint.org.uk/Default.aspx.

[9] Q Technology (Group) Company Limited. *Q Tech*. 2014.
http://qtechglobal.com/product_info_1.html.

[10] 3GVision. *i-nigma Camera Phone Barcode Reader SDK*. 2012-2017.
http://www.i-nigma.com/qr-barcode-reader-sdk.html.

[11] Vision Smarts. *VS Barcode Reader*. 2009-2017.
http://www.visionsmarts.com/products/vs-barcode-reader.html.

[12] Cognex Corporation. *Barcode Scanner SDK*. 2017.
https://manateeworks.com/barcode-scanner-sdk.

[13] Droidla Limited. *QR Droid zapper*. 2014.
http://qrdroid.com/.

[14] SCANDIT. *SCANDIT*. 2016.
http://www.scandit.com/.

[15] Jeff Brown. *ZBar bar code reader*. 2007-2010.
http://zbar.sourceforge.net/index.html.

[16] Inc. GitHub. *ZBar*. 2017.
https://github.com/ZBar/ZBar.

[17] Inc. GitHub. *Official ZXing ("Zebra Crossing") project home*. 2017.
https://github.com/zxing/zxing.

[18] United statespostal service. *Intelligent Mail Barcode 4-State SPECIFICATION*. 4/20/2015.
https://ribbs.usps.gov/intelligentmail_mailpieces/documents/tech_guides/USPSB3200IntelligentMailBarcode4State.pdf.

[19] Pavel Strachota. *Teorie signálu pro počítačovou grafiku*. 19. února 2015. http://saint-paul.fjfi.cvut.cz/base/sites/default/files/POGR/POGR2/02.teorie_signalu.pdf.