VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

VÍCEDIMENSIONÁLNÍ AUTOMATY A JEJICH APLIKACE V UMĚNÍ

MULTIDIMENSIONAL FINITE AUTOMATA AND THEIR APPLICATIONS IN ART

BAKALÁRSKA PRÁCA
BACHELOR'S THESIS

AUTOR PRÁCE **MÁRIO GAŽO**
AUTHOR

VEDOUCÍ PRÁCE **prof. RNDr. ALEXANDER MEDUNA, CSc.**
SUPERVISOR

BRNO 2021

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav informačních systémů (UIFS) Akademický rok 2020/2021

Zadání bakalářské práce

Student: **Gažo Mário**
Program: Informační technologie
Název: **Vícedimensionální automaty a jejich aplikace v umění**
Multi-Dimensional Automata and Their Applications in Art
Kategorie: Teoretická informatika

Zadání:

1. Dle instrukcí vedoucího se seznamte s různými vícedimensionální automaty a jejich jazyky.
2. Zaveďte nové verze těchto automatů dle instrukcí vedoucího.
3. Studujte vlastnosti těchto automatů a jazyků dle instrukcí vedoucího. Porovnejte jejich sílu s jinými automaty.
4. Dle instrukcí vedoucího se seznamte s výtvarným uměním, např koláže od Jiřího Koláře či Ladislava Nováka. Aplikujte automaty z bodu 2 v tomto umění dle instrukcí vedoucího.
5. Implementujte aplikace navržené v bodě 4.
6. Zhodnoťte dosažené výsledky. Diskutujte další vývoj projektu.

Literatura:

- Rozenberg, G., Salomaa, A. (eds.): Handbook of Formal Languages, Volume 1-3, Springer, 1997, ISBN 3-540-60649-1

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 a 3.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Meduna Alexander, prof. RNDr., CSc.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 12. května 2021
Datum schválení: 27. října 2020

Abstrakt

Táto práca sa zaoberá prepojením teoretickej informatiky a výtvarného umenia. Demonštruje silu a možnosti viacdimenzionálnych, najmä dvojdimenzionálnych, automatov v umení a to tak, že ich aplikuje na koláž, kde riadi pohyb jednotlivých prvkov.

Abstract

This thesis deals with interconnection of theoretical computer science and fine arts. It demonstrates power and possibilities of multidimensional, mainly two-dimensional, finite automata in fine arts by applying them to collage and controlling individual elements movement.

Kľúčové slová

konečný automat, viacdimenzionálny konečný automat, teoretická informatika, výtvarné umenie, koláž

Keywords

finite automata, multidimensional finite automata, theoretical computer science, fine arts, collage

Citácia

GAŽO, Mário. *Vícedimensionální automaty a jejich aplikace v umění*. Brno, 2021. Bakalárska práca. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce prof. RNDr. Alexander Meduna, CSc.

Vícedimensionální automaty a jejich aplikace v umění

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením prof. RNDr. Alexandra Meduny, CSc. Uviedol som všetky literárne pramene, publikácie a ďalšie zdroje, z ktorých som čerpal.

.....
Mário Gažo
3. mája 2021

Poďakovanie

Týmto by som sa chcel poďakovať svojmu vedúcemu prof. RNDr. Alexandrovi Medunovi, CSc., za cenné rady, pomoc pri vypracovaní, inšpiráciu a vedenie mojej práce, veľmi si to vážim. Ďalej som vďačný rodine, priateľom a kolegom za psychickú podporu. 92101

Obsah

Kapitola 1

Úvod a cieľ

V tejto práci sa budem zaoberať aplikáciou viacdimenzionálnych automatov v umení. Jedná sa o prepojenie teoretickej informatiky a výtvarného umenia, kde teoretickú informatiku zastupuje viacdimenzionálny automat a umenie umelecká technika, ktorá zodpovedá počtu dimenzií automatu.

Keďže nenadväzujem na inú prácu, ktorá by sa touto tématikou zaoberala, musím začať preštudovaním dvodimenzionálnych automatov, práce ktoré by na túto nadväzovali, by mohli preskúmať aj automaty vyšších dimenzií. Tým pádom musím zvoliť umeleckú techniku adekvátnu počtu dimenzií automatu. V tomto prípade som zvolil koláž, tá je tvorená podkladom a prvkami na ňom. Tieto prvky sa budú pohybovať, vytvárať rôzne efekty a to na základe dvodimenzionálneho automatu, ktorý bude tento pohyb riadiť. Efektov bude viacero a bude sa medzi nimi dať prechádzať na základe vstupu diváka.

V nasledujúcich kapitolách sa dopracujem k výsledku, predtým si ale musím definovať viacero pojmov, vytvoriť tak teoretický základ automatu, ten následne implementovať a otestovať.

Najprv definujem jednodimenzionálne konečné automaty, teda nedeterministický a deterministický a popíšem ich spoločné a rozdielne vlastnosti. Taktiež definujem jazyky ktoré prijímajú. Ďalej definujem obecný dvodimenzionálny konečný automat a známe typy tohto automatu.

Následne sa budem zaoberať kolážou ako umeleckou technikou, ale aj jej históriou. Ako táto technika vznikla, kedy a kto ju zdokonalil, princípom tejto umeleckej techniky a umelcami, ktorý ju tvorili. Pre účely ilustrácie predstavím aj niektoré známe koláže.

Ďalej vytvorím námet pre nový typ dvojdimenzionálneho automatu, ktorý bude vhodný pre kombinovanie s kolážou, bude riadiť pohyb jej prvkov. Tento automat bude vychádzať z obecného dvojdimenzionálneho automatu.

Potom navrhnem spôsob, akým bude tento automat implementovaný. Vyberiem si vhodné technológie, vytvorím abstraktný model automatu, ktorý rozdelím na moduly. Tie sú tvorené dátami ktoré nesú a metódami, ktoré s týmito dátami pracujú. Na základe týchto modulov bude premietaný výstup automatu, teda samotné nové umelecké dielo ktoré vznikne skombinovaním automatu a koláže.

Napokon aplikujem tento už implementovaný automat na niekoľkých ozajstných umeleckých dielach. Vyberiem diela z rôznych umeleckých období a demonštrujem na nich jednotlivé efekty automatu.

Nakoniec zhodnotím svoje úsilie. Posúdim k akým záverom som sa dopracoval, popíšem priestor pre ďalšie práce, ktoré by mohli vzniknúť v nadväznosti na túto.

Kapitola 2

Konečné automaty

V tejto kapitole sa budem zaoberať známymi jednodimenzionálnymi a dvojdimenzionálnymi konečnými automatmi a jazykmi ktoré prijímajú. Platí, že dimenzia automatu je adekvátna jazyku s ktorým automat pracuje, ktorý prijíma. Jednodimenzionálny automat prijíma a spracúva reťazce symbolov, teda slová. Dvojdimenzionálny automat prijíma dvojdimenzionálny jazyk, ten tvorí dvojdimenzionálne pole symbolov, teda obraz.

2.1 Jednodimenzionálny konečný automat

Obecný jednodimenzionálny automat definujeme ako päticu $M = (Q, \Sigma, R, s, F)$ [10], kde platí, že:

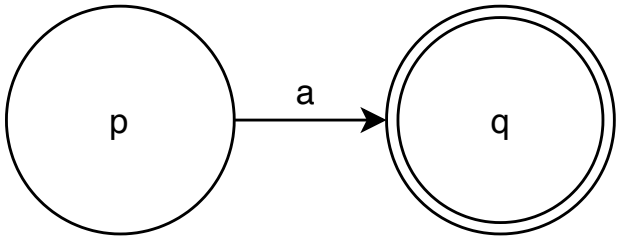
Q je konečná množina stavov automatu

Σ je vstupná abeceda, ide o neprázdnu a konečnú množinu symbolov

R je konečná množina pravidiel v tvare: $pa \to q$ $(p, q \in Q \wedge a \in \Sigma)$

s je počiatočný stav automatu $(s \in Q)$

F sú koncové stavy automatu $(F \in Q)$



Obr. 2.1: $pa \to q$

Aktuálny stav automatu závisí od prijatej postupnosti symbolov z vstupnej abecedy. Na začiatku čítania je aktuálnym stavom počiatočný stav, ďalej sa na základe prijatého symbolu aktuálnym stavom stáva stav, ktorý určí vhodné pravidlo z množiny pravidiel R. Všetky postupnosti symbolov po ktorých prijatí sa automat dokáže dostať z počiatočného stavu do koncového stavu tvoria reťazce patriace do jazyka definovaného týmto automatom. Jazyk ktorý definuje tento automat označujeme $L(M)$.

2.1.1 Nedeterministický konečný automat

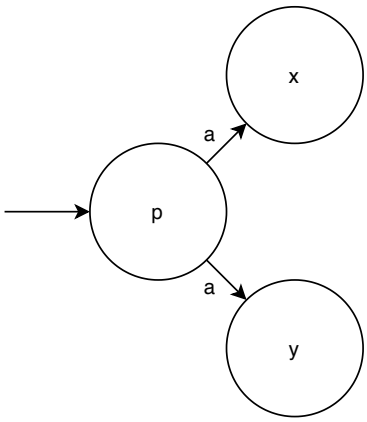
[11] Ak má obecný konečný automat jednu z nasledujúcich vlastností, tak je považovaný za nedeterministický. Ide o vlastnosti, ktoré zapríčiňujú, že nie je jasné, do akého stavu sa má po prečítaní ďalšieho symbolu automat dostať. V takomto prípade je nasledujúci stav automatu daný iným spôsobom, ktorý môže byť rôzny, napríklad náhodným výberom.

Nejednoznačné pravidlá Môže sa stať, že medzi pravidlami konečného automatu budú také pravidlá, ktoré spôsobujú nedeterminizmus tým, že z určitého stavu sa po prečítaní symbolu môže automat dostať do viacerých stavov súčasne. Nie je jednoznačné, aký stav sa má stať aktuálnym stavom automatu po prečítaný určitého symbolu v určitom stave. 2.2 Zo stavu 'p' sa po prečítaní symbolu 'a' môže automat dostať do stavu 'x' aj 'y'.

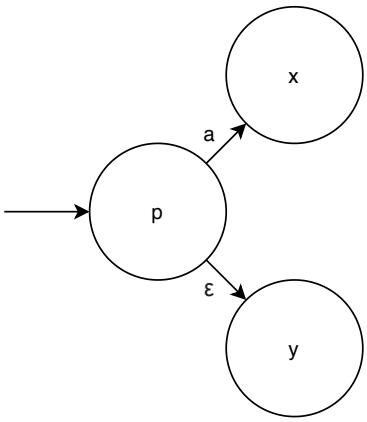
ε-prechody Tieto prechody značia také prechody automatu, pri ktorých aj bez prijatia symbolu z abecedy môže nastať prechod z aktuálneho stavu do iného. Nie je možné jednoznačne určiť kedy sa vykonajú, alebo či budú mať prednosť pred prechodmi podmienenými prijatím symbolu. 2.3 Zo stavu 'p' sa automat môže dostať do stavu 'y' svojvoľne, nezávisle od toho či prijme symbol 'a'.

Nedostupné stavy Konečný automat môže obsahovať aj stavy, do ktorých sa z počiatočného stavu nie je možné dostať po prečítaní akéhokoľvek reťazca tvoreného symbolmi vstupnej abecedy. 2.4 Stav 'x' je nedostupný.

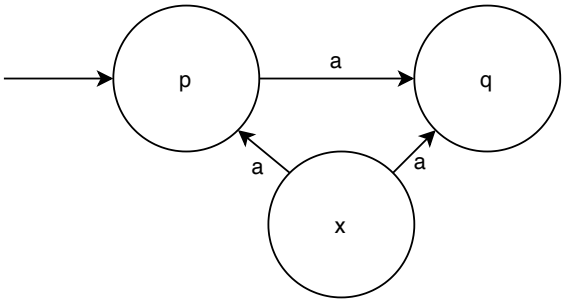
Neukončujúce stavy Neukončujúci stav je taký stav do ktorého sa dokáže automat dostať prečítaním určitého reťazca. Z tohto stavu sa však automat nedokáže dostať do iného. Tým že stav nie je ukončujúci nie je zrejmé, či prijatý reťazec do jazyka ktorý tento automat definuje patrí alebo nie. 2.5 Stav 'x' je neukončujúci.



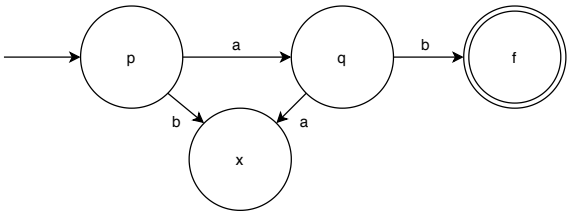
Obr. 2.2: Nejednoznačné pravidlá



Obr. 2.3: ε-prechod



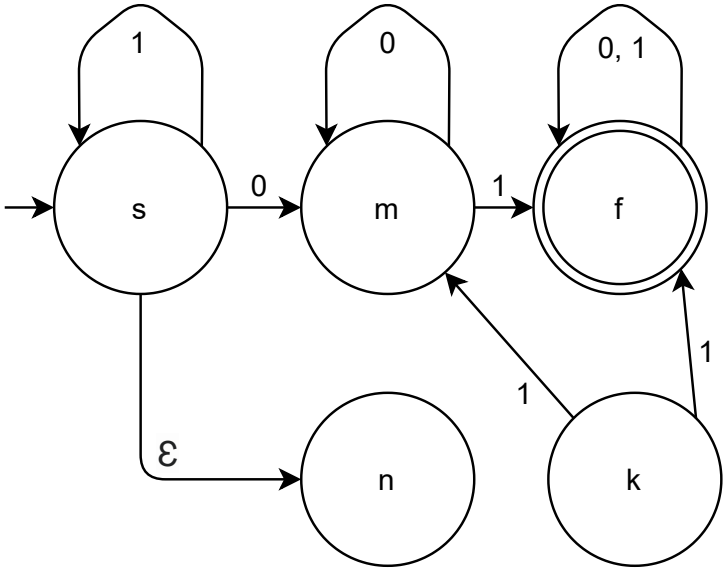
Obr. 2.4: Nedostupný stav



Obr. 2.5: Neukončujúci stav

Príklad 2.1.1 *Majme konečný automat* $M = (Q, \Sigma, R, s, F)$*, kde:*

- $Q = \{s, m, n, k, f\}$
- $\Sigma = \{0, 1\}$
- $R = \{s1 \to s, s0 \to m, s \to n, k1 \to m, k1 \to f\, m0 \to m, m1 \to f, f0 \to f, f1 \to f\}$
- $s = \{s\}$
- $F = \{f\}$



Obr. 2.6: Nedeterministický konečný automat

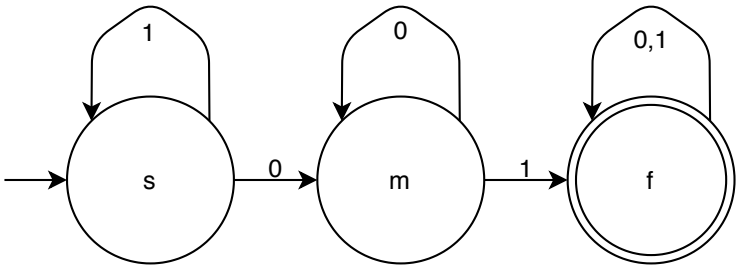
Automat sa skladá z piatich stavov a prijíma jazyk ktorého reťazce sú tvorené symbolmi '0' a '1'. Platí, že každý prijatý reťazec obsahuje podreťazec '01'. Zároveň platí, že automat je nedeterministický, nakoľko má všetky črty nedeterminizmu. Všetky prechody zo stavu 'k' reflektujú na prijatie symbolu '1', to ich robí nejednoznačné. Pravidlo $s \to n$ je nejednoznačné ε-pravidlo, na základe neho sa automat dokáže dostať zo stavu 's' do stavu 'n' bez prijatia symbolu. Stav 'k' je nedostupným stavom, neexistuje cesta z počiatočného stavu 's' do 'k'. Stav 'n' je neukončujúci.

2.1.2 Deterministický konečný automat

[10]Každý nedeterministický konečný automat M sa dá zrekonštruovať na deterministický konečný automat M'. Rekonštrukcia pozostáva z odstránenia vyššie spomenutých vlastností. Tieto automaty si sú ekvivalentné, popisujú rovnaký jazyk. Platí, že $L(M) = L(M')$.

Príklad 2.1.2 *Majme konečný automat* $M = (Q, \Sigma, R, s, F)$*, kde:*

- $Q = \{s, m, f\}$
- $\Sigma = \{0, 1\}$
- $R = \{s1 \to s, s0 \to m, m0 \to m, m1 \to f, f0 \to f, f1 \to f\}$
- $s = \{s\}$
- $F = \{f\}$



Obr. 2.7: Deterministický konečný automat

Nakoľko tento automat nemá ani jednu z uvedených čŕt nedeterministického automatu, jedná sa o deterministický automat. Skladá sa z troch stavov a prijíma jazyk ktorého reťazce sú tvorené symbolmi 0 a 1, pričom platí, že každý reťazec obsahuje podreťazec 01.

2.2 Dvojdimenzionálny konečný automat

Rozšírením jednodimenzionálneho automatu o schopnosť čítať nie len lineárny reťazec znakov, ale pole znakov a o možnosť čítať vstup vo viacerých smeroch vzniká dvojdimenzionálny konečný automat [13]. Jazyk s ktorým pracuje je teda obraz o rozmeroch m x n. Okraje tohto obrazu sú tvorené hraničným symbolom, ktorý nie je súčasťou vstupnej abecedy.

2.2.1 Obecný dvodimenzionálny automat

Obecný dvojdimenzionálny automat definujeme ako šesticu $M = (Q, \Sigma, \delta, q_0, q_A, q_R)$, kde platí, že:

Q je konečná množina stavov

Σ je vstupná abeceda ($\# \notin \Sigma$ je hraničný symbol)

δ je tranzitívna funkcia

q_0 je počiatočný stav automatu ($q_0 \in Q$)

q_A je akceptačný stav automatu ($q_A \in Q$)

q_R je odmietajúci stav automatu ($q_R \in Q$)

#	#	#	#	#	#
#	1,1	1,2	...	1, n - 1	#
#	2,1	2,2	...	2, n - 1	#
#	...	...	...	...	#
#	m - 1, 1	m - 1, 2	...	m - 1, n - 1	#
#	#	#	#	#	#

Obr. 2.8: Dvojdimenzionálny automat s rozmermi $m \times n$

Pre identifikáciu buniek vrámci automatu budem používať súradnicový systém v ktorom narozdiel od bežnej matematickej notácie, kde bod so súradnicami $(0, 0)$ je vľavo dole, je tento bod vľavo hore 2.8. Je to výhodné, pretože pri vykresľovaní objektov vo väčšine grafických knižníc bežných programovacích jazykov je potrebné zadať ľavý horný bod objektu. Všetky existujúce obrazy nad abecedou sa označujú $\Sigma^{m \times n}$.

Nasledujúci stav automatu je určený tranzitívnou funkciou ktorej vstupom je aktuálny stav a pole symbolov. Výstupom je nasledujúci stav automatu. Práve na základe tejto funkcie sa rozlišujú jednotlivé typy dvodimenzionálnych automatov. Definujem viacero známych typov.

2.2.2 Celulárny automat

Je to typ dvojdimenzionálneho automatu ktorý je tvorený poľom buniek. To sú elementárne časti automatu, ktoré udržujú jednu z možných hodnôt. Môžu susediť napríklad s tromi, štyrmi či šiestimi ďalšími bunkami. Nasledujúca hodnota bunky je určená hodnotami buniek v jej okolí. Okolie bunky tvoria bunky s ktorými buď bezprostredne alebo sprostredkovane susedí. Poznáme viacero typov okolí [12].

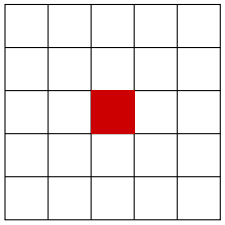
Prázdne 2.9 Bunka mení svoju hodnotu nezávisle od okolitých buniek.

Von Neumannovo 2.10 Toto okolie tvoria bunky v štyroch základných smeroch (vpravo, vľavo, hore, dole).

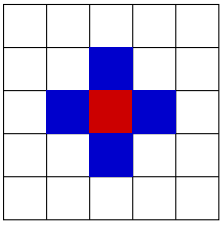
Moorovo 2.11 Nasledujúci stav bunky je závislý od ôsmych buniek v štyroch základných smeroch a štyroch diagonálnych smeroch (vpravo hore, vpravo dole, vľavo hore, vľavo dole)

Rozšírené Von Neumannovo 2.12 Von Neumannovho okolie sa dá zväčšovať zvýšením polomeru.

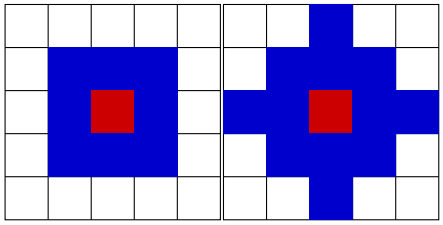
Rozšírené Moorovo 2.13 Moorovo okolie je možné rozšíriť o ďalšie bunky zvýšením polomeru.



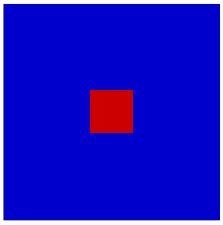
Obr. 2.9: Prázdne



Obr. 2.10: Von Neumannovo



Obr. 2.11: Moorovo



Obr. 2.12: Rozšírené VN

Obr. 2.13: Rozšírené M

Príklad 2.2.1 *Majme celulárny automat o rozmeroch 7x7 s určitým počiatočným stavom. Uvažujme Moorovo okolie. V automate platia určité pravidlá. Tieto pravidlá sú špecifické pre Game Of Life [1], typický celulárny automat:*

1 *Bunka môže byť v stave 0 alebo 1.*

2 *Bunka je v stave 1, ak sú dve alebo tri okolité bunky v stave 1, bunka ostáva v stave 1, inak sa bunka dostáva do stavu 0.*

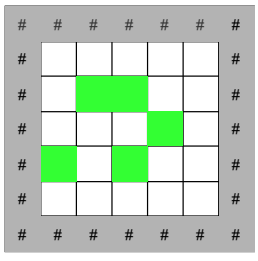
3 *Bunka je v stave 0, ak je obklopená tromi bunkami v stave 1, dostáva sa do stavu 1.*

t = 0 2.14 V počiatočnom stave je päť buniek v stave 1 (zelená), ostatné sú v stave 0 (bez farby). Okolo každej bunky je vytvorené Moorovo okolie a sú spočítané nasledujúce hodnoty buniek.

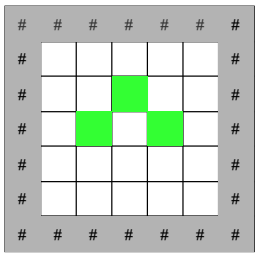
t = 1 2.15 Po kroku automatu sa jedna bunka zmení stav na 1, dve zostanú v stave 1 a ostatné sa dostanú do stavu 0.

t = 2 2.16 Ďalej nastane situácia, kde sa jedna bunka dostane do stavu 1, jedna zostane v stave 1 a ostatné sa zo stavu 1 dostanú do stavu 0.

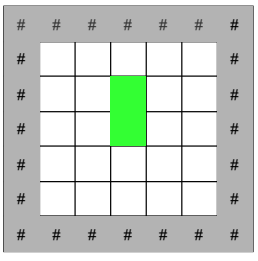
t = 3 2.17 Nakoniec sa všetky bunky dostanú do stavu 0 a automat ostane prázdny.



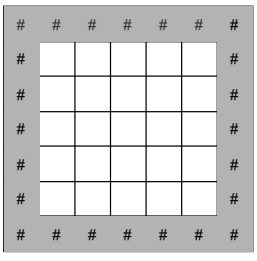
Obr. 2.14: t = 0



Obr. 2.15: t = 1



Obr. 2.16: t = 2



Obr. 2.17: t = 3

2.2.3 Štvorcestný konečný automat

[9] Štvorcestný konečný automat (4FA) je dvodimenzionálny variant konečného stavového akceptoru. Platí, že

$$\delta : ((Q \setminus \{q_a, q_r\}) \times \Sigma) \to 2^{(Q \times \{L,R,U,D\})}$$

kde $\{L, R, U, D\}$ znamená ľavo, pravo, hore a dole. Takýto automat prijíma jazyk, ktorý tvorí podmnožina polí znakov z množiny všetkých polí, ktoré môžu vzniknúť nad abecedou Σ. Polia znakov ktoré prijíma, musia z počiatočného stavu spĺňať určitú sekvenciu hodnôt, ktorú udá tranzitívna funkcia δ.

Príklad 2.2.2 *Majme štvorcestný automat o rozmeroch 7x7. V tomto automate platí niekoľko pravidiel:*

1 *Každý prvok nesie číselnú informáciu.*

2 *Ak bunka nesie hodnotu 0, posunie sa nahor a zvýši hodnotu.*

3 *Ak bunka nesie hodnotu 1, posunie sa doprava a zvýši hodnotu.*

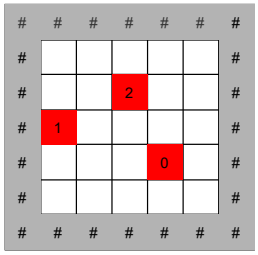
4 *Ak bunka nesie hodnotu 2, posunie sa nadol a zvýši hodnotu.*

5 *Ak bunka nesie hodnotu 3, posunie sa doľava a nastaví hodnotu na 0.*

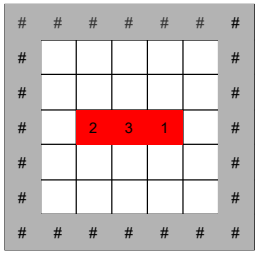
t = 0 2.18 V počiatočnom stave je automat pokrytý tromi prvkami s hodnotami 0, 1 a 2. Po vykonaní kroku automatu sa každý prvok posunie na základe vyššie stanovených pravidiel do určitého smeru a aktualizuje svoju hodnotu. Prvok 0 nahor a nastaví hodnotu na 1, prvok 1 doprava a nastaví hodnotu na 2 a prvok 2 nadol a nastaví hodnotu na 3.

t = 1,2,3 2.19 2.20 2.21 Obdobne v nasledujúcich krokoch.

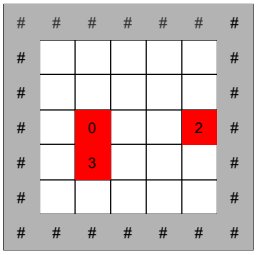
Takýmto spôsobom vzniká automat, ktorého prvky sa pohybujú po štyroch bunkách a po štyroch krokoch sa dostanú do rovnako stavu, čiže v čase t = 4 by opäť nasledoval počiatočný stav.



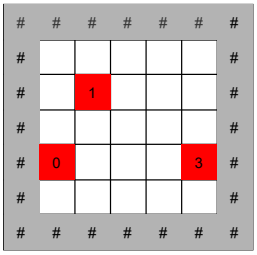
Obr. 2.18: t = 0



Obr. 2.19: t = 1



Obr. 2.20: t = 2



Obr. 2.21: t = 3

2.2.4 Teselačný konečný automat

[9, 4] Teselačný konečný automat (2OTA) je obmedzený typ celulárneho automatu. Vstupom je dvodimenzionálne pole buniek x, kde $x(i, j)$ je bunka so súradnicami (i, j).

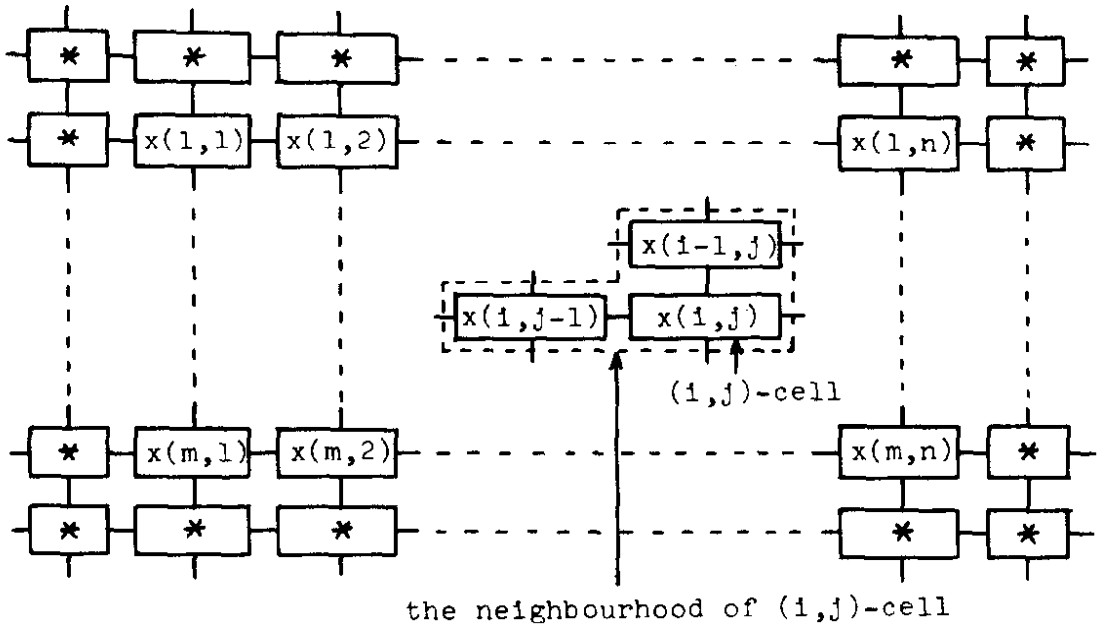
Platí, že:

- Pre každú bunku $x(i,j)$ je zostrojený konečný automat a vypočítaná nasledujúca hodnota na základe symbolov na pozíciách $(i-1,j)$ a $(i,j-1)$ 2.22

$$q_{(i,j)}(t+1) = \delta(q_{(i,j)}(t), q_{(i-1,j)}(t), q_{(i,j-1)}(t), a)$$

  kde a je hodnota bunky (i,j)

- V k-tom kroku automatu, každá bunka so súradnicami (i,j), kde $i+j-1=k$, zmení svoju hodnotu na základe už zmienenej funkcie. To znamená, že v každom kroku sa vykonajú zmeny hodnôt buniek automatu na príslušnej diagonále.



Obr. 2.22: Okolie bunky (i,j) [4]

Príklad 2.2.3 *Majme online teselačný automat o rozmeroch* $7x7$.

V každom kroku automatu je spracovaná množina buniek pre ktoré platí $k = i+j-1$, kde k je poradie kroku a i, j sú súradnice bunky.

1. krok 2.23 V prvom kroku je ovplyvnená len jedna bunka so súradnicami (1,1).
k = 1
i = 1, j = 1: $1 = 1+1-1$.

2. krok 2.24 V nasledujúcom kroku sú spracované dve bunky so súradnicami (2,1) a (1,2).
k = 2
i = 1, j = 2: $2 = 1+2-1$
i = 2, j = 1: $2 = 2+1-1$

3. krok 2.25 V ďalšom kroku bunky so súradnicami (1,3), (2,2) a (3,1).
k = 3
i = 1, j = 3: $3 = 1+3-1$
i = 2, j = 2: $3 = 2+2-1$
i = 3, j = 1: $3 = 3+1-1$

4. krok 2.26 V poslednom zobrazenom kroku (1,4), (2,3), (3,2) a (4,1).

$k = 4$

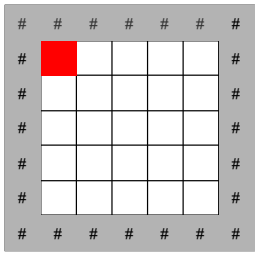
$i = 1, j = 4$: $4 = 1 + 4 - 1$

$i = 2, j = 3$: $4 = 2 + 3 - 1$

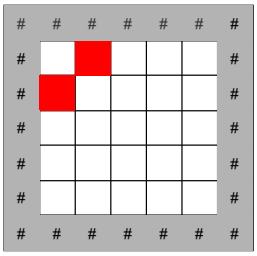
$i = 3, j = 2$: $4 = 3 + 2 - 1$

$i = 4, j = 1$: $4 = 4 + 1 - 1$

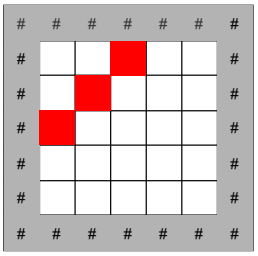
V každom kroku sú obslúžené bunky na diagonálne, ktorá je daná poradím kroku.



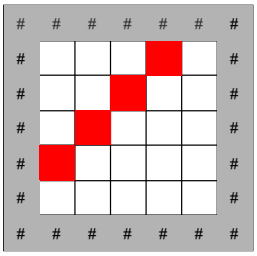
Obr. 2.23: $k = 1$



Obr. 2.24: $k = 2$



Obr. 2.25: $k = 3$



Obr. 2.26: $k = 4$

Kapitola 3

Koláž

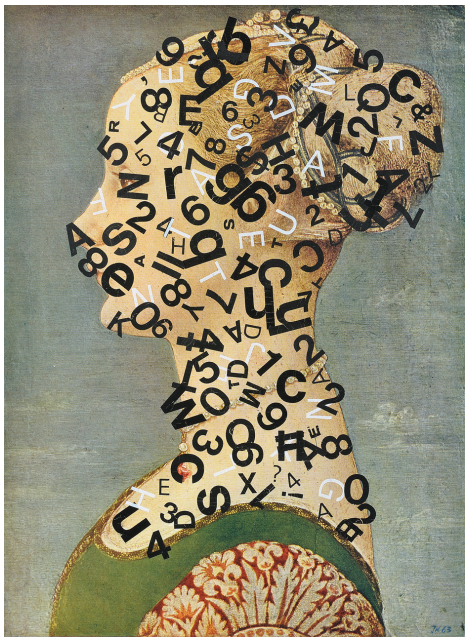
Meno tejto umeleckej techniky je odvodené z francúzskeho *collage*, teda "lepiť". [15] Je to umelecká technika, ktorá sa zdokonalila v šesťdesiatych a sedemdesiatych rokoch dvadsiateho storočia vrámci avantgardného prúdu kubizmus. Využíva sa hlavne vo vizuálnom umení, je charakterizovaná kombinovaním médií a umeleckých výtvorov takým spôsobom, že vznikne nové vnímanie diela ako celku. Zvyčajne sa vytvára nanášaním menších prvkov, ako sú útržky z novín, vystrihnuté symboly, ikony a podobne, na väčší, nemenný povrch, napríklad výkres alebo úplne iné umelecké dielo. Takýto spôsobom dostáva obraz nový, viac priestorový rozmer, ktorý je práve pre kubizmus typický.

Najdôležitejším umelcom, ktorý sa venoval tvorbe koláží bol predstaviteľ kubizmu, španielsky umelec *Pablo Picasso (1881 - 1973)*. Hoci bol pôvodne zo Španielska, väčšinu života prežil vo Francúzsku, kde aj tvoril. Narozdiel od dovtedajšej koncepcie koláže, ktorá používala hlavne primitívne geometrické tvary ako kruhy a štvorce, v jeho koncepcií koláže používal prvky ako písmená, noty, fragmenty slov a snažil sa vytvoriť z pôvodného obrazu viac hmatateľný objekt. Príklady jeho diel sú nižšie 3.1 3.2.

Obr. 3.1: Bottle of Vieux Marc, Glass, Guitar and Newspaper, 1913 [8]

Obr. 3.2: A glass and a bottle of Suze, 1912 [7]

Najvýraznejším českým umelcom, ktorý tvoril koláže bol spisovateľ a maliar *Jiří Kolář (1914 - 2002)*. Tým, že vytvoril mnohé nové techniky tvorby koláže sa zaradil k najvýznamnejším českým umelcov dvadsiateho storočia. Jeho diela sú zobrazené nižšie 3.3 3.4. Práve dielo Popisný portrét bolo jednou z inšpirácii riešenia tejto bakalárskej práce. Jeho podklad tvorí obraz dámy z profilu, prvky sú rôzne natočené čierno-biele písmená a čísla, pokrývajú oblasť hlavy a krku.



Obr. 3.3: Popisný portrét, 1963 [5]

Obr. 3.4: Ruka, 1964 [6]

Ďalším dôležitým pojmom je digitálna koláž. Jedná sa o novú techniku tvorby koláže. Celý proces nanášania menších prvkov na väčší obraz je digitalizovaný za pomoci výpočtovej techniky.

Kapitola 4

Nový typ konečného automatu

Za účelom prepojenia dvojdimenzionálneho automatu s výtvarným umením som navrhol nový typ dvojdimenzionálneho automatu. Tento automat vychádza z obecného automatu, ktorý som definoval vyššie. Ostatné špecifickejšie automaty sú definované svojou tranzitívnou funkciou δ, tá udáva akým spôsobom sú menené hodnoty buniek a celkový stav automatu. Tento automat je špecifický tým, že jeho tranzitívna funkcia nie je fixná, ale predstavuje skôr zoznam funkcií, medzi ktorými môže užívateľ interaktívne prepínať. Je teda možné aby bol každý krok vykonaný iným spôsobom. Zmenou tranzitívnej funkcie sa mení aj vizuálny výstup.

4.1 Náhodnosť

V automate funguje určitý prvok náhody, ktorý sa zavádza vo viacerých situáciách. V každom prípade je náhodnosť rovnomerná, každá možnosť má rovnakú pravdepodobnosť. Jedná sa o diskrétne rovnomerné rozdelenie pravdepodobnosti.

x	1	2	...	n
$p(x)$	$\frac{1}{n}$	$\frac{1}{n}$	...	$\frac{1}{n}$

Obr. 4.1: Rozdelenie pravdepodobnosti

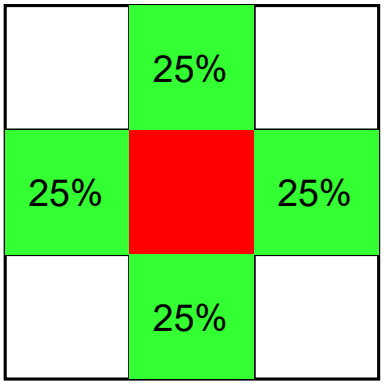
Platí že:

- x je možná hodnota náhodnej premennej
- $p(x)$ je pravdepodobnostná funkcia, vracia pravdepodobnosť zvolenia zadanej možnosti x
- $\sum_{i=1}^{n} p(x_i) = \frac{1}{n} * n = 1$

4.1.1 Smer pohybu prvku

V automate je viacero situácii v ktorých je potrebné k určitej bunke náhodne vybrať jednu z jej susedných buniek. Za týmto účelom sa vytvorí množina možných smerov pohybu, z ktorej sa jeden vyberie. Pravdepodobnosť výberu každého zo smerov je rovnaká.

Príklad 4.1.1 *Majme dvojdimenzionálny automat s jedným prvkom.*



Obr. 4.2: Smery pohybu

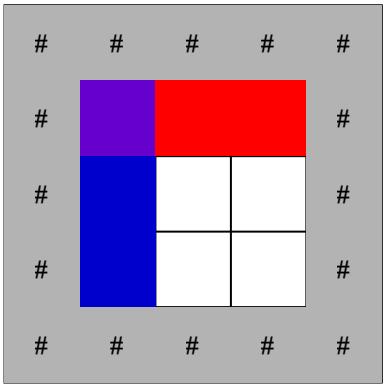
Platí, že:

- pre každý zo štyroch možných smerov pohybu je pravdepodobnosť 25%
- $\sum_{i=1}^{n} p(x_i) = 0.25 + 0.25 + 0.25 + 0.25 = 1$

4.1.2 Náhodná bunka

Aby výber bunky bol náhodný a zároveň táto náhodnosť zostala rovnomerná, je potrebné premietnuť rovnomerné rozloženie do dvoch dimenzií. To prebehne tak, že sa náhodne zvolí riadok automatu, výber každého riadka má rovnakú pravdepodobnosť. Ďalej sa zvolí náhodný stĺpec automatu, rovnako pravdepodobnosť výberu každého z nich je rovnaká. Tým vznikne dvojica náhodných súradníc, ktoré označujú bunku automatu.

Príklad 4.1.2 *Majme dvojdimenzionálny automat o rozmeroch 5x5.*



Obr. 4.3: Náhodná bunka

Najprv je zvolený náhodný riadok (červená), následne náhodný stĺpec (modrá). Tam kde sa prekryjú tieto dve hodnoty vzniká jediná náhodná bunka (fialová).
Platí, že:

- pravdepodobnosť výberu každého stĺpca je $\frac{1}{5} = 20\%$
- pravdepodobnosť výberu každého riadku je $\frac{1}{5} = 20\%$
- pravdepodobnosť výberu každej bunky je $\frac{1}{5*5} = 4\%$

4.2 Počiatočný stav

Pri každom spustení automatu vzniká pole buniek. Tieto bunky sú náhodne obsadené určitým počtom prvkov. V automate sa vygeneruje náhodná bunka, skontroluje sa, či nie je obsadená iným prvkom a ak nie, bunku obsadí daný prvok. To sa opakuje, kým nie je vygenerovaný daný počet prvkov.

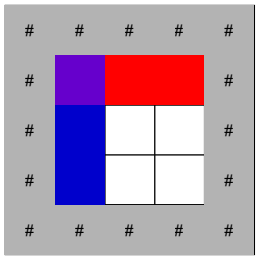
Príklad 4.2.1 *Majme dvojdimenzionálny automat o rozmeroch 5x5.*

t = 0 4.4 Prvý prvok je jednoznačne náhodne umiestnený.

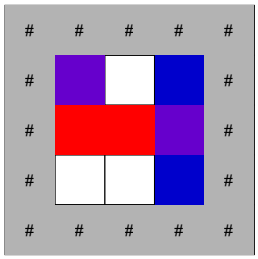
t = 1 4.5 Druhý prvok je podobne zasadený do náhodne zvolenej bunky.

t = 2 4.6 Pri treťom prvku nastáva komplikácia, náhodne zvolená bunka je už obsadená iným prvkom. V takom prípade sa opakuje proces generovania náhodnej bunky.

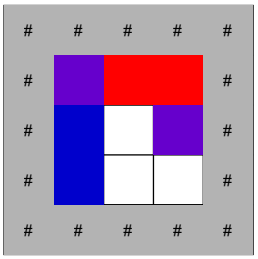
t = 3 4.7 V nasledujúcom kroku sa aj tretí prvok umiesti do náhodnej, prázdnej bunky.



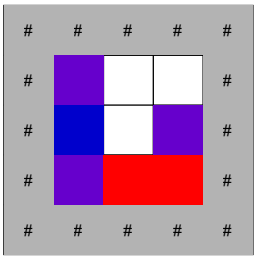
Obr. 4.4: t = 0



Obr. 4.5: t = 1



Obr. 4.6: t = 2



Obr. 4.7: t = 3

4.3 Efekty

Automat je definovaný viacerými efektami, ktoré je schopný vytvárať. V tejto časti ich pomenujem a definujem. Každý z nich predstavuje inú tranzitívnu funkciu, fungujúcu na inom princípe. Každý z efektov je vnútorne reprezentovaný číselnou hodnotou, stlačením tohto čísla na klávesnici sa dynamicky mení efekt za iný.

1 Štvorcestný

2 Štvorcestný na základe typu

3 Výmena na základe typu

4 Rozptýlenie

0 Pauza

4.3.1 Štvorcestný

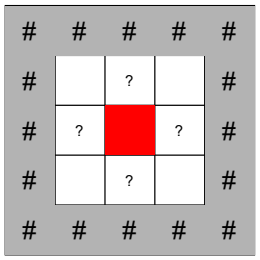
Jedná sa o základný efekt, spúšťa sa spolu s automatom. Je založený na pohybe prvkov do štyroch základných smerov náhodným spôsobom. V každom kroku sú obslúžené všetky prvky automatu. Prvok sa v náhodnom poradí pokúsi o premiestnenie do bunky v každom zo štyroch základných smerov. Ak je bunka v jednom z takýchto smerov voľná a zároveň nie je hraničná (neobsahuje znak #, ktorý nepatrí do abecedy), je vykonaný posun. Takto je každý prvok buď posunutý, alebo ak nebola ani jedna bunka v žiadnom zo smerov vhodná, prvok ostáva na svojom mieste. Prvok môže byť vo viacerých situáciách, na základe ktorých sa vyberá ďalší smer pohybu. V popise obrázkov sú možné smery pohybu.

Voľný pohyb 4.8 Bunka má pre svoj pohyb voľné všetky štyri bunky v každom zo smerov. V nasledujúcom kroku sa môže prvok presunúť do ktoréhokoľvek z nich.

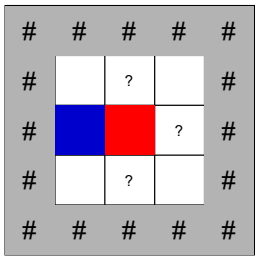
Blokovanie prvkom 4.9 Bunka je v jednom z možných smerov pohybu blokovaná iným prvkom. Prvok sa môže posunúť len do troch smerov, obdobne by sa nemohol posunúť aj do iných smerov, ak by bol blokovaný iným prvkom. Červený prvok sa nemôže presunúť do bunky naľavo, táto bunka je blokovaná modrým prvkom.

Blokovanie hranicou 4.10 V takomto prípade je prvok blokovaný hranicou automatu, ktorú netvoria platné bunky. Na ilustračnej situácii, je prvok taktiež blokovaný aj iným prvok v jednom zo smerov. Prvok sa tým pádom určite pohne smerom nadol.

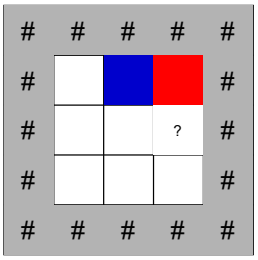
Úplné blokovanie 4.11 V poslednej situácii má prvok blokovaný každý smer pohybu buď iným prvkom alebo hranicou. Prvok nemá kam migrovať, preto ostáva na rovnakej pozícii.



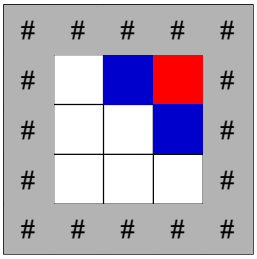
Obr. 4.8: L, R, U, D



Obr. 4.9: R, U, D



Obr. 4.10: D



Obr. 4.11: {}

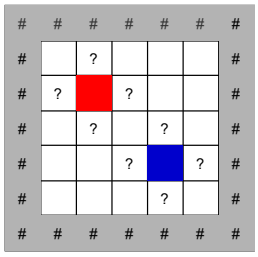
Príklad 4.3.1 *Majme dvodjimenzionálny automat o rozmeroch 7x7 s dvoma prvkami. Uvažujme, že aktívnym efektom je štvorcestný efekt.*

t = 0 4.12 V počiatočnom stave sú v automate dva prvky rôznych typov. Každý z nich sa môže presunúť do všetkých štyroch možných smerov. Možné smery sú naznačené otáznikmi. Smer pohybu je zvolený náhodne, červený nadol, modrý nahor.

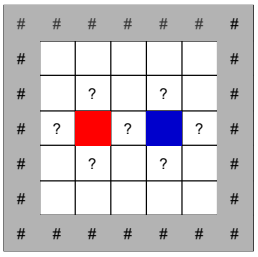
t = 1 4.13 Vzniká kolízia možných smerov pohybu týchto prvkov v prostrednej bunke. Ak by sa v nasledujúcej iterácií presunul jeden z prvkov do tejto bunky, ten druhý by už mal na výber len tri smery.

t = 2 4.14 Červený prvok presunutý do kolíznej bunky a modrá o bunku nahor.

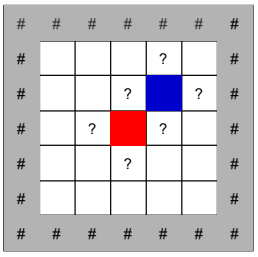
t = 3 4.15 Vznikajú dve kolízne bunky. V poslednej situácii sa oba prvky posunuli doprava. Modrý sa dostal až k hranici automatu, jeho nasledujúci pohyb bude možný len v troch smeroch.



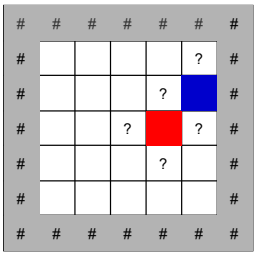
Obr. 4.12: t = 0



Obr. 4.13: t = 1



Obr. 4.14: t = 2



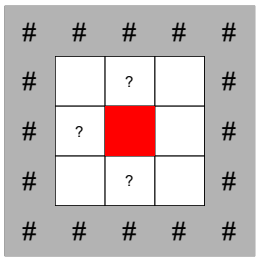
Obr. 4.15: t = 3

4.3.2 Štvorcestný na základe typu

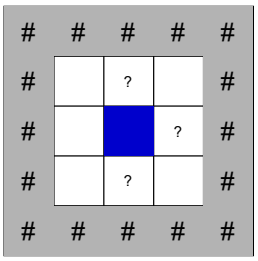
Rovnako ako štvorcestný efekt, aj tento efekt v každom kroku prechádza každý prvok a jeho susedné bunky v základných smeroch, aby zvolil kam sa posunie. Rozdiel spočíva v tom, že tento efekt pri výbere možných smerov pohybu berie ohľad aj na typ prvku. Určitý typ prvku sa nemôže pohnúť do jedného zo smerov. Týmto spôsobom sa postupne začnú prvky spoločného typu zarovnávať určitým smerom. Takéto situácie môžme vidieť na príkladoch nižšie. V popise obrázkov sú možné smery pohybu obsluhovaného prvku.

Voľný pohyb s reštrikciou Jeden zo smerov pohybu nie je možný pre typ prvku. Keď sú dva prvky na rovnakej pozícii v automate, ale nie sú rovnakého typu, ich množina možných smerov pohybu nie je rovnaká 4.16 4.17.

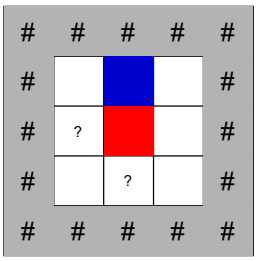
Blokovanie prvkom Podobne ako v prípade pôvodného štvorcestného efektu, priľahlý prvok spôsobuje blokovanie smeru a tým pádom zmenu v množine voľných smerov 4.18 4.19.



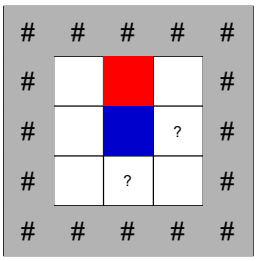
Obr. 4.16: L, U, D



Obr. 4.17: R, U, D



Obr. 4.18: L, D



Obr. 4.19: R, D

Príklad 4.3.2 *Majme dvojdimenzionálny automat o rozmeroch 7x7 s dvomi prvkami rôznych typov. Uvažujme, že aktívnym efektom je štvorcestný na základe typu.*

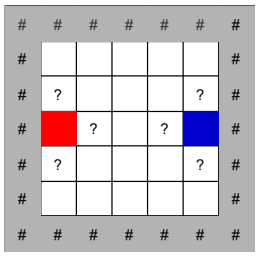
t = 0 4.20 V počiatočnom stave má automat dva prvky, každý iného typu. Oba sa môžu pohnúť do troch smerov. Modrý prvok sa posunie doľava, červený nadol. Po tejto translácií sa opäť vygenerujú množiny možných smerov pohybu.

t = 1 4.21 Oba prvky sa môžu pohnúť do troch smerov, modrý sa nemôže pohnúť doprava, červený doľava. Následne sa červený prvok pohne doprava a modrý doľava.

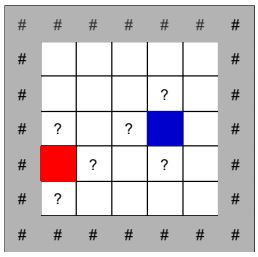
t = 2 4.22 Tak vzniknú dve kolízne bunky. Taktiež je možné vidieť, že červený prvok sa nemôže pohnúť doľava, preto nie je naznačený ako možný smer.

t = 3 4.23 Nakoniec sa oba prvky opäť posunú do jedného z možných smerov.

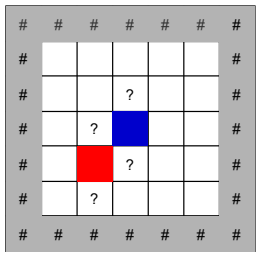
Červený prvok sa nikdy nemôže posunúť doľava a modrý doprava. Keby automat operoval ďalej, tak by sa každý z prvkov postupne dostal na hranicu, kde by sa posúval len nahor a nadol.



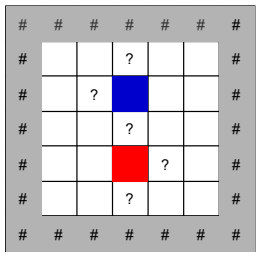
Obr. 4.20: t = 0



Obr. 4.21: t = 1



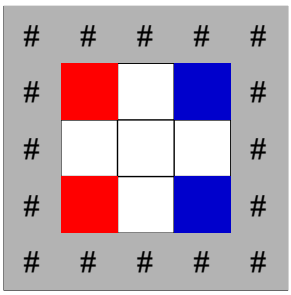
Obr. 4.22: t = 2



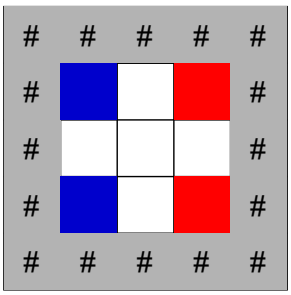
Obr. 4.23: t = 3

4.3.3 Výmena na základe typu

Podobne ako doteraz definované efekty, aj v tomto prípade automat prechádza jednotlivé prvky. V každom kroku vyberie v náhodnom poradí jeden z prvkov, skontroluje jeho typ a následne vyberie ďalší náhodný prvok iného typu. Potom vymení pozície týchto dvoch prvkov a odstráni ich množiny prvkov. Keďže v každom momente je počet prvkov určitého typu rovnaký, postupne sa vymení každý prvok s iným. Na obrázkoch 4.24 a 4.25 je štvorica prvkov, dve dvojice prvkov určitých typov, ktoré sa po kroku automatu medzi sebou vymenia.



Obr. 4.24: Pred



Obr. 4.25: Po

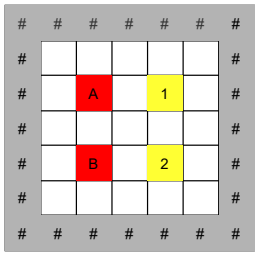
Príklad 4.3.3 *Majme dvojdimenzionálny automat o rozmeroch 7x7 so štyrmi prvkami. Uvažujme, že aktívnym efektom je výmena na základe typu.*

t = 0 4.26 V počiatočnom stave je automat pokrytý štyrmi prvkami, dva z nich sú typu písmená (červená), symboly A a B, ostatné dva sú čísla (žltá), symboly 1 a 2.

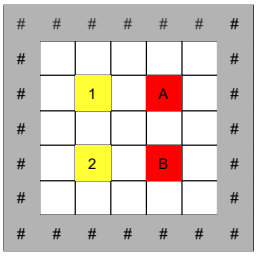
t = 1 4.27 Po iterácii automatu sú vymenené obe skupiny prvkov, symbol A za symbol 1 a symbol B za symbol 2.

t = 2 4.28 Potom sú znova vymenené obe skupiny, tentokrát symbol A za symbol 2 a symbol B za symbol 2.

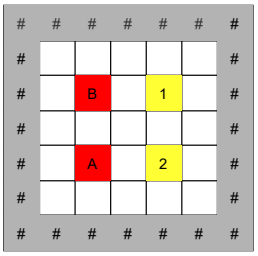
t = 3 4.29 V poslednom kroku sa opäť vymieňajú obe skupiny prvkov a opäť rovnakým spôsobom ako v prvom kroku.



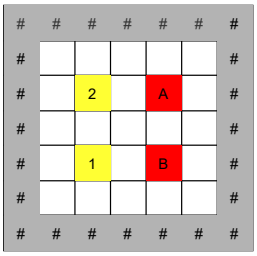
Obr. 4.26: t = 0



Obr. 4.27: t = 1



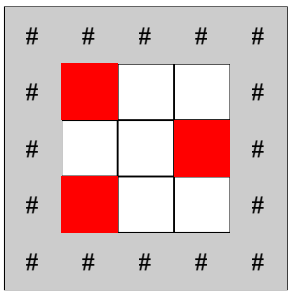
Obr. 4.28: t = 2



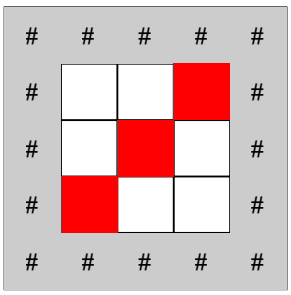
Obr. 4.29: t = 3

4.3.4 Rozptýlenie

Posledným pravým efektom automatu je náhodný rozptyl prvkov. Je definovaný náhodným rozmiestňovaním prvkov po poli. Neexistuje žiadna korelácia medzi rozmiestnením prvkov v jednotlivých iteráciách. Automat pokrývajú tri prvky 4.30. Po iterácii automatu 4.31 všetky prvky náhodne zmenia svoju polohu. Počet prvkov na poli sa nemení.



Obr. 4.30: Pred



Obr. 4.31: Po

Príklad 4.3.4 *Majme dvojdimenzionálny automat o rozmeroch 7x7 so šiestimi prvkami. Uvažujme, že aktívnym efektom je rozptýlenie.*

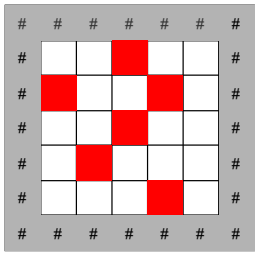
t = 0 4.32 V počiatočnom stave je automat náhodne pokrytý šiestimi prvkami.

t = 1 4.33 V následnom stave je automat taktiež pokrytý šiestimi prvkami, no v rozdielnej konfigurácii. Tento stav nie je nijak závislý od predchádzajúceho stavu.

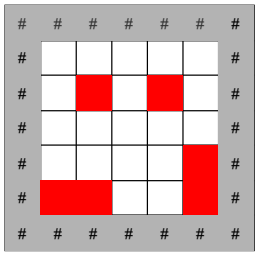
t = 2 4.34 Po ďalšej iterácii automatu sú na automate prvky v rovnakom počte, opäť v rozdielnej polohe a opäť nezávisle od predchádzajúceho stavu.

t = 3 4.35 V poslednom stave je znova vytvorené originálne rozmiestnenie prvkov po automate.

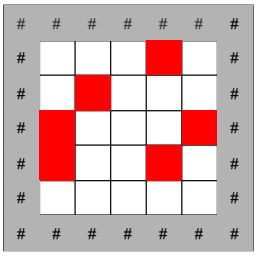
Žiaden zo stavov automatu nie je závislý od predchádzajúceho stavu. V každom stave automatu je počet a typ prvkov rovnaký.



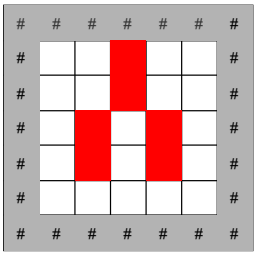
Obr. 4.32: t = 0



Obr. 4.33: t = 1



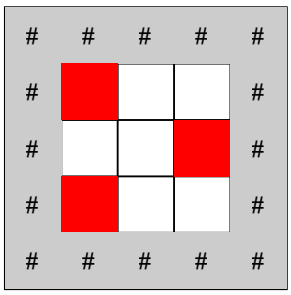
Obr. 4.34: t = 2



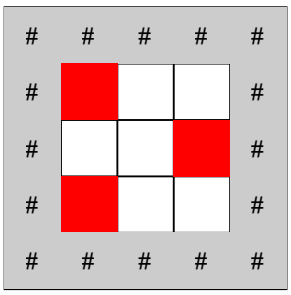
Obr. 4.35: t = 3

4.3.5 Pauza

Pre úplnosť spomínam aj prázdny efekt. Ten sa využíva pre pozastavenie automatu. Keď je aktívny, prvky nemenia svoju polohu. Žiaden prvok automatu nemení svoje súradnice, nevytvárajú sa žiadne nové prvky ani žiadne existujúce nezanikajú. Z počiatočného stavu 4.36 automatu po iterácii automatu vždy vznikne rovnaký stav ako bol počiatočný 4.37.



Obr. 4.36: Pred



Obr. 4.37: Po

Kapitola 5

Implementácia

Po definovaní teoretického základu a z neho vyplývajúceho nového typu dvojdimenzionálneho automatu sa budem venovať samotnému technickému vypracovaniu. Pomenujem technológie, ktoré som sa rozhodol použiť. Načrtnem architektúru riešenia, rozdelenie na moduly, ktoré nesú dáta a metódy ktoré s nimi operujú. Takýmto spôsobom sa riešia podproblémy, ktoré tvoria celok, nové umelecké dielo. Zároveň zdôrazním časti kódu, ktoré sú najdôležitejšie a špecifikujem akým spôsobom korešpondujú s teoretickým základom.

5.1 Technológie

Pri práci s grafickým rozhraním je na výber množstvo technológií, programovacích jazykov a im vlastných knižníc pre tvorbu grafických rozhraní. Zvážil som napríklad jazyk Java [1] a knižnicu JavaFX [2], či jazyk C++ [3] a knižnicu Qt [4]. Následne som sa rozhodol pre nasledujúce technológie.

Python3.7 [5] Skriptovací jazyk v ktorom je vypracovaná celá praktické časť. Volím ho aj preto, že je dostatočne intuitívny, nebude potreba popisovať algoritmy pseudo kódom.

Tkinter [6] Ako mnoho iných jazykov, aj Python má svoju vstavanú knižnicu pre tvorbu grafických rozhraní. Tú som použil na otváranie okna, vykresľovanie a pohyb prvkov.

Pillow / PIL [7] Niektoré možnosti, ktoré som potreboval využiť pri vypracovaní som nenachádzal vo vstavaných knižniciach. Siahol som aj po externej knižnici. Použil som ju na načítanie obrázku a na jeho vykreslenie do okna.

[1] https://www.java.com/en/
[2] https://openjfx.io/
[3] https://www.cplusplus.com/
[4] https://www.qt.io/
[5] https://www.python.org/
[6] https://docs.python.org/3/library/tkinter.html
[7] https://pillow.readthedocs.io/en/stable/

5.2 Spustenie

Aplikácia sa spúšťa cez príkazový riadok s jedným argumentom. Tento argument udáva relatívnu alebo absolútnu cestu k obrázku v digitálnej forme, ktorý bude tvoriť podklad digitálnej koláže. Obrázok musí byť vo formáte .png alebo .jpg.

```
> python main.py obraz
```

5.3 Vyčíslenia

Pre úplnosť spomínam, že pri implementovaní som vytvoril viacero vyčíslení pre lepšiu vnútornú organizáciu kódu. Všetko sú to triedy, ktoré dedia z vstavanej triedy Enum.

Effects FOUR_WAY, FOUR_WAY_TYPE, SWAP_TYPE, SCATTER, PAUSE

Ways RIGHT, LEFT, UP, DOWN

ElementTypes LETTER, NUMBER, DEBUG

5.4 Trieda Base

Pre podklad automatu som vytvoril hlavnú triedu aplikácie. Trieda dedí z tkinter triedy Tk, slúži ako koreňový element na ktorý sú vykresľované ďalšie prvky. Vytvorením inštancie tejto triedy, je otvorené okno s rozmermi načítaného obrázku a s podkladom, ktorý tvorí práve tento obrázok.

5.4.1 Konštruktor

```
def __init__(self, path: str)
```

Pre vytvorenie inštancie je potrebné zadať parameter path. Ten predstavuje cestu k obrázku, ktorý je načítaný. Následne je vytvorené plátno s rozmermi obrázku, vložené do koreňového elementu a je naň nanesený obraz. Ďalej sú z rozmerov obrázku vypočítané ostatné potrebné parametre pre chod automatu.

5.4.2 Dáta

Vrámci triedy sú zapuzdrené viaceré kľúčové dáta týkajúce sa automatu, jeho funkcií, proporcií a zobrazovania.

SQUARE_SIZE je konštanta, ktorá reprezentuje približnú šírku a výšku bunky, experimentovaním s väčšími a menšími obrazmi som dospel k hodnote 30 pixelov

UPDATES je počet krokov automatu za sekundu

DEBUG je interná konštanta, ak je nastavená na True, tak sú na obraz zakreslené hranice buniek, hraničné symboly a neprejavuje sa prvok náhodnosti pri vykreslovaní

canvas je plátno automatu, na ktoré je vykreslený obraz a prvky

elements je pole prvkov automatu

width je šírka obrazu v pixeloch

height je výška obrazu v pixeloch

cells_y je výška automatu v bunkách

```
cells_y = round(height / SQUARE_SIZE)
```

cells_x je šírka automatu v bunkách, ich počet je závislý od šírky automatu a pomeru šírky a výšky obrazu

```
cells_x = cells_y * (width / height)
```

cell_h je výška bunky

```
cell_h = round(height / cells_y)
```

cell_w je šírka bunky

```
cell_w = round(width / cells_x)
```

effect je aktuálne aktívny efekt, základným efektom je štvorcestný efekt

```
effect = Effects.FOUR_WAY
```

5.4.3 Metódy

Pre riadenie automatu a prácu s dátami obsahuje trieda sadu funkcií.

def keydown(self, event: tkinter.Event) Obsluhuje stlačenie klávesy. V char položke objektu event obsahuje hodnotu stlačenej klávesy. Mapuje numerické hodnoty na jednotlivé efekty. Stlačením numerickej hodnoty sa zmení aktuálny efekt a v nasledujúcom kroku bude vykonaný.

```
# Mapa efektov na zadane znaky
self.effect = {
    '1': Effects.FOUR_WAY,
    '2': Effects.FOUR_WAY_TYPE,
    '3': Effects.SWAP_TYPE,
    '4': Effects.SCATTER,
    '0': Effects.PAUSE
}[event.char]
```

def cell_ok(self, x: int, y: int) Kontrola bunky identifikovanej relatívnymi súradnicami (x, y). Vracia True ak je možné, aby sa do tejto bunky presunul nejaký prvok. Bunka je nevhodná ak je obsadená iným prvkom, je hraničnou bunkou alebo je mimo automatu.

```
# Bunka je mimo automatu alebo je na hranici
if not (0 < x < self.cells_x - 1 and 0 < y < self.cells_y - 1):
    return False

# Bunka obsahuje prvok
```

```
for e in self.elements:
    if e.x == x and e.y == y:
        return False
return True
```

def get_free_cell(self) Vráti dvojicu relatívnych súradníc (x, y). Tieto súradnice patria náhodnej bunke automatu, ktorá nie je obsadená prvkom.

```
random_x = randrange(1, self.cells_x - 1)
random_y = randrange(1, self.cells_y - 1)
while not self.cell_ok(random_x, random_y):
    random_x = randrange(1, self.cells_x - 1)
    random_y = randrange(1, self.cells_y - 1)
return random_x, random_y
```

def generate_set(self, count: int, elem_type: ElementTypes) Generovanie prvkov v počte count a typu elem_type. Pri každej iterácii sa vygeneruje náhodná voľná bunka automatu. Nový prvok je vložený do tejto bunky a je pridaný do poľa elements.

```
for _ in range(count):
    # Nájdi prázdnu bunku, vlož do nej prvok a ulož ho
    random_x, random_y = self.get_free_cell()
    element = Element(self, random_x, random_y, elem_type)
    self.elements.append(element)
```

def generate_elements(self, count: int) Generovanie písmen a čísel. Počet prvkov každého typu je daný parametrom count.

```
generate_set(count, ElementTypes.LETTER)
generate_set(count, ElementTypes.NUMBER)
```

Písmen a čísiel je počas celého behu automatu rovnaký počet. Počet prvkov nemôže byť fixný, pretože malé obrazy by mohli byť zaplnené a na veľkých by prvky zanikali, preto musí byť tento počet relatívny k rozmerom obrazu. Experimentovaním som prišiel k záveru, že je najviac efektívne keď prvky v každom kroku automatu zapĺňajú približne dvadsať percent buniek obrazu.

```
# 10% buniek pre oba typy prvkov
round(self.cells_y * self.cells_x * 0.1)
```

def movement(self) Na základe zvoleného efektu zavolá adekvátnu tranzitívnu funkciu, s výnimkou efektu PAUSE. Následne naplánuje vykonanie samej seba, tým sa vykoná krok automatu. Takto vznikne nekonečný cyklus, ktorý predstavuje obsluhu automatu.

```
{
    Effects.FOUR_WAY: self.move_4_ways,
    Effects.FOUR_WAY_TYPE: self.move_4_ways,
    Effects.SWAP_TYPE: self.swap,
    Effects.SCATTER: self.scatter,
```

```
}[self.effect]()
after(round(1000 / self.UPDATES), movement)
```

def try_{smer}(self, element: Element) Pokúsi sa prvok posunúť do jedného zo štyroch základných smerov (LEFT, RIGHT, UP, DOWN). Ak je pohyb možný, funkcia vracia True a vykoná pohyb, v opačnom prípade vracia None a prvok pohyb nevykoná.

```
# Napríklad pohyb doprava, ostatne smery fungujú analogicky
if self.cell_ok(element.x + 1, element.y):
    return element.move(Ways.RIGHT)
```

def get_options(self, element: Element) Na základe typu prvku a aktívneho efektu vygeneruje pole možných smerov pohybu prvku. Náhodne premieša toto pole a vráti ho. Používa sa pre rozlíšenie medzi štvorcestným efektom a jeho obdobou na základe typu.

```
if self.effect is Effects.FOUR_WAY_TYPE:
    if e.elem_type == ElementTypes.LETTER:
        options = range(0, 3)
    else:
        options = range(1, 4)
else:
    options = range(0, 4)
return sample(options, len(options))
```

def move_4_way(self) Tranzitívna funkcia pre štvorcestný efekt (FOUR_WAY) a štvorcestného efektu na základe typu (FOUR_WAY_TYPE). Prechádza prvky automatu, pre každý nich vygeneruje pole možných smerov pohybu. Následne prejde toto pole a pokúsi sa pohyby vykonať. Pri prvom možnom pohybe sa pohyb vykoná. Takýmto spôsobom je buď vykonaný pohyb alebo, ak nie je možné vykonať ani jeden smer pohybu, prvok zostane na svojom mieste. Po obslúžení všetkých prvkov je vykonaný krok automatu.

```
# Mapa smerov
ways = {
    0: lambda x: try_right(x),
    1: lambda x: try_up(x),
    2: lambda x: try_down(x),
    3: lambda x: try_left(x),
}

for e in self.elements:
    options = self.get_options(e)
    # Pokus o pohyb
    for option in options:
        if ways[option](e):
            break
```

def swap(self) Tranzitívna funkcia efektu výmeny na základe typu (SWAP_TYPE). Vytvorí kópiu pola prvkov, následne prechádza kópiu. Vyberá si dva náhodné prvky kým nenájde také, ktoré majú rozdielny typ. Keď na také narazí, vymení ich pozíciu a odstráni ich z kópie pola prvkov. Toto sa opakuje kým sa kópia nevyprázdni. Vrámci kroku si každý prvok vymení svoju pozíciu s iným prvkom rozdielneho typu.

```
copy = self.elements.copy()
while len(copy) > 0:
    # Dva náhodné prvky
    random = sample(copy, 2)
    if random[0].elem_type is not random[1].elem_type:
        # Výmena a odstránenie
        random[0].swap(random[1])
        copy.remove(radnom[0])
        copy.ramove(random[1])
```

def scatter(self) Tranzitívna funkcia efektu rozptýlenia (SCATTER). Prejde všetky prvky automatu a nastaví ich súradnice na neplatné, aby neobsadzovali bunky na ktoré budú prvky náhodne kladené. Následne sa vygenerujú náhodné platné bunky a prvky sa do nich nasádzajú.

```
# Zneplatnenie prvkov
for e in self.elements:
    e.x, e.y = -1, -1

# Rozptýlenie
for e in self.elements:
    x, y = self.get_free_cell()
    e.draw(x,y)
```

def grid(self) Pomocná funkcia pre ladenie aplikácie, môže byť použitá aj pri ilustrovaní fungovania automatu. Vykreslí na obraz vertikálne a horizontálne čiary, čím zobrazí bunky automatu. Do buniek na hraniciach automatu vloží nehybné prvky s neplatným znakom #.

```
# Generovanie vertikálnych čiar a hraničných prvkov
# Analogicky sú generované aj horizontálne čiary
for i in range(self.cells_x):
    self.canvas.create_line(
        self.cell_w * i, 0,
        self.cell_w * i, self.height)
    Element(self, i, 0, ElementTypes.DEBUG)
    Element(self, i, cells_x - 1, ElementTypes.DEBUG)
```

5.5 Trieda Element

Touto triedou sú reprezentované jednotlivé prvky digitálnej koláže. Generujú sa na konkrétnu koláž, na určitú horizontálnu a vertikálnu súradnicu. Pohyb prvkov po koláži zabezpečuje sada metód. Na základe tvorivého podnetu od Jiřího Koláře a jeho diela

Popisný portrét, sú v aplikácií pohyblivými prvkami písmená a čísla. Prvok je vždy určitého typu.

LETTER veľké písmená od A po Z

NUMBER čísla od 0 po 9

DEBUG neabecedný znak # pre hraničné bunky

5.5.1 Konštruktor

```
def __init__(self, base: Base, x: int, y: int, elem_type: ElementTypes)
```

Prvok patrí konkrétnej koláži, ktorá je referovaná parametrom base. Je kladený do konkrétnej bunky, ktorá je daná svojimi relatívnymi súradnicami (x, y). Parameter elem_type určuje typ prvku a teda aj aký symbol bude prvok na koláži predstavovať.

5.5.2 Dáta

Každý prvok obsahuje dáta na základe ktorých je vykresľovaný a pohybovaní po koláži.

x, y sú relatívne súradnice bunky, ktorú prvok obsadzuje

w, h sú šírka a výška bunky automatu

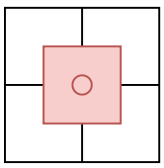
elem_type je typ prvku

canvas je plátno na ktorom je prvok nanesený

id je identifikácia prvku vrámci plátna

5.5.3 Metódy

def draw(self, x: int = -1, y: int = -1) Zakreslenie prvku na plátno na základe relatívnych súradníc prvku a rozmeroch bunky. Vrámci funkcie nastáva prepočet relatívnych súradníc na absolútne súradnice stredu bunky v pixeloch. Aby prvky neboli len sterilne presúvané z bunky do bunky, pridal som určitý prvok náhodnosti pri vykresľovaní. K vertikálnej súradnici sa pripočítava štvrť výšky bunky vynásobená náhodným číslom s rovnomerným rozložením na intervale (-1, 1), rovnako s horizontálnou súradnicou a šírkou bunky. V strede bunky tak vznikne oblasť, v ktorej sa vyskytuje stred prvku. Táto oblasť predstavuje štvrť plochy bunky 5.1. Relatívne súradnice prvku sú konzistentné, náhodnosť nastáva len pri kreslení.



Obr. 5.1: Bunka a oblasť so stredom prvku

```
# Súradnice sú zadané
if x != -1 and y != -1:
    self.x, self.y = x, y
# Prepočet súradníc, zohľadnenie náhodnosti
self.canvas.coords(self.id,
    self.x * self.w + self.w/2 + self.w/4 * uniform(-1,1),
    self.y * self.h + self.h/2 + self.h/4 * uniform(-1,1))
```

def move_{smer}(self) Funkcie meniace relatívne súradnice prvku, podľa toho akým smerom sa má pohnúť (right, left, up a down).

```
# Napriklad pohyb doprava
self.x += 1
```

def move(self, way: Ways) Obsluhuje pohyb do jedného zo základných smerov. Parameter way určuje smer, ktorým sa má prvok pohnúť. Prvok dostane súradnice novej bunky a prekreslí sa.

```
# Mapa smerov na funkcie pohybu
{
    Ways.RIGHT: move_right,
    Ways.LEFT: move_left,
    Ways.UP: move_up,
    Ways.DOWN: move_down
}[way]()
self.draw()
```

def swap(self, e: Element) Stará sa o výmenu pozícií dvoch prvkov, konkrétne o výmenu prvku z ktorého je táto metóda volaná a prvku e, ktorý je predaný ako parameter. Prvkom sú vymenené relatívne súradnice a sú opätovne vykreslené.

```
# Výmena súradníc
self.x, self.y, e.x, e.y = e.x, e.y, self.x, self.y

# Opätovné vykreslenie prvkov
self.draw()
e.draw()
```

Kapitola 6

Testovanie

V tejto pasáži aplikujem implementovaný nový typ konečného automatu na konkrétne obrazy. Vybral som moje tri obľúbené obrazy z rôznych období. Sú rôznych rozmerov a rôznych pomerov výšky a šírky, prvý je širší, druhý štvorcový a tretí vyšší. Pre každý z nich zobrazím počiatočný stav v režime pre ladenie, na obraz sú vykreslené hranice buniek a hraničné bunky sú obsadené neabecedným znakom #. Ďalej zachytím počiatočný stav bez režimu ladenia. Na prvom z nich demonštrujem všetky efekty.

6.1 Stvorenie Adama

Prvý obraz, pre ktorý som sa rozhodol je fragment notoricky známeho výjavu *Stvorenie Adama*. Autorom je jeden z majstrov renesančného obdobia v Taliansku, *Michelangelo Buonarroti (1475-1564)*. Jedná sa o fresku ktorá zdobí strop Sixtínskej kaplnky vo Vatikáne. Celé dielo zobrazuje Boha a prvého muža Adama ako sa len tak tak dotýkajú, ale nie úplne. Detail na tento dotyk 6.1 tvorí podklad pre testovanie. Tento obraz si vyberám aj preto, že jeho šírka je pomerne väčšia ako jeho výška.

Obr. 6.1: Stvorenie Adama (detail), 1508-1512 [2]

6.1.1 Parametre

Výška obrazu 328 pixelov

Šírka obrazu 728 pixelov

Výška automatu $\frac{328}{30} \doteq 11$ buniek

Pomer šírky a výšky $\frac{728}{328} \doteq 2.22$

Šírka automatu $11 * 2.22 \doteq 24$ buniek

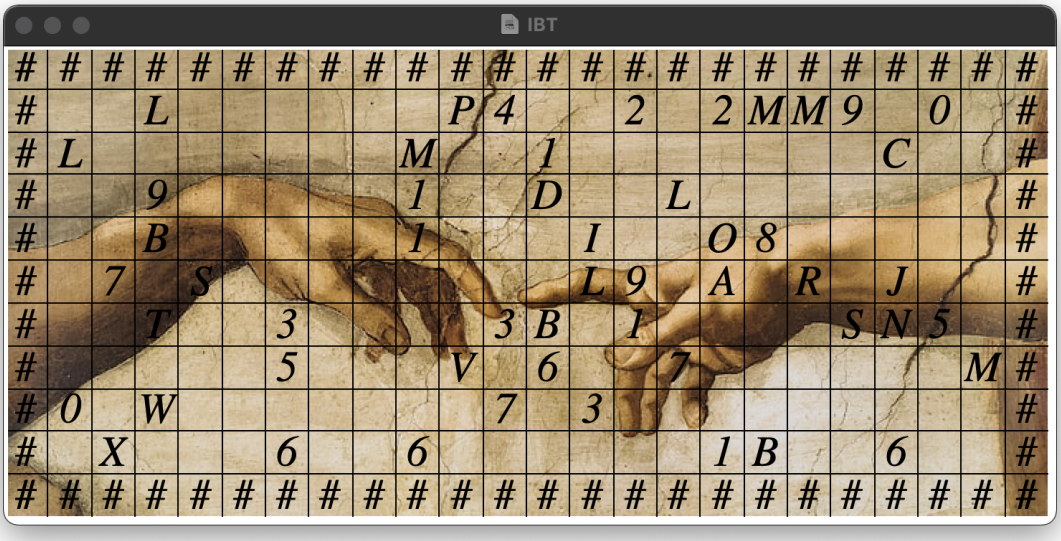
Výška bunky $\frac{328}{11} \doteq 29.81$ pixelov

Šírka bunky $\frac{728}{24} \doteq 30.33$ pixelov

Počet buniek $24 * 11 = 264$ buniek

Počet prvkov jedného typu $264 * 0.1 \doteq 26$ prvkov

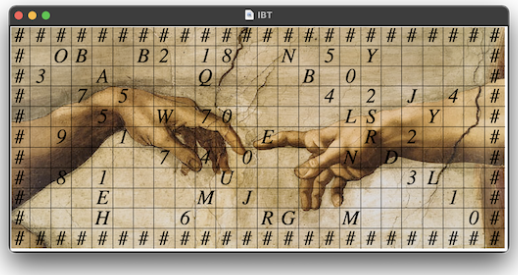
Automat má aj s hraničnými oblasťami dvadsaťštyri buniek na šírku a jedenásť buniek na výšku. Každá bunka má približne tridsať pixelov na výšku a tridsať pixelov na šírku. Zo všetkých dvestošesťdesiatštyri vygenerovaných buniek je prvkami obsadených dvadsaťšesť písmenami a dvadsaťšesť číslami 6.2.



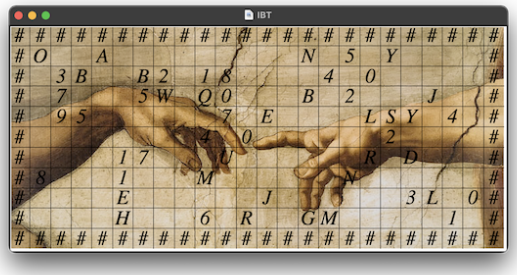
Obr. 6.2: Režim ladenia

6.1.2 Štvorcestný efekt

Prvky automatu sa náhodne pohybujú do štyroch základných smerov. Prvok O (vľavo) sa posunul doľava, prvok 3 (vľavo) doprava, prvok 9 (vľavo) nahor a prvok 1 (vpravo) nadol. Porovnanie stavov automatu je zobrazené nižšie na obrázkoch 6.3 a 6.4.

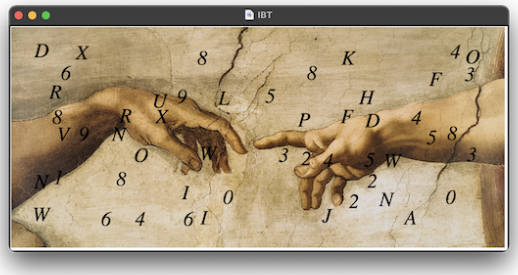


Obr. 6.3: t = 0



Obr. 6.4: t = 1

Podobne aj pre automat bez pomocných čiar a s prvkom náhodnosti. Prvky v čase t = 0 6.5 v porovnaní s prvkami v čase t = 49 6.6 zmenili svoju polohu a dostali sa to inej časti automatu. Napríklad prvok W sa dostal zľava viac do stredu.

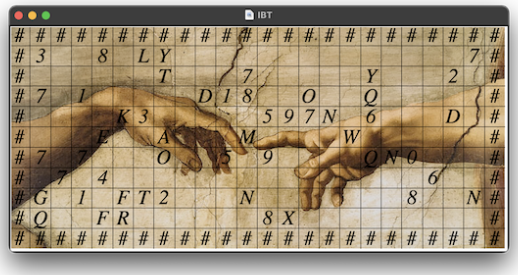


Obr. 6.5: t = 0

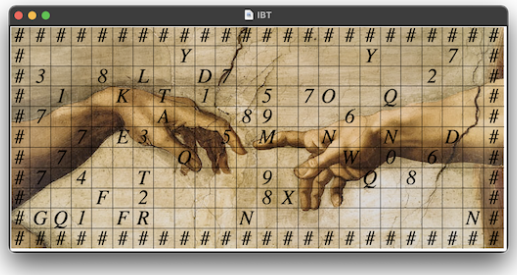
Obr. 6.6: t = 49

6.1.3 Štvorcestný efekt na základe typu

Prvky automatu sa náhodne pohybujú do štyroch smerov no s reštrikciami na základe typu prvku. Písmená sa nikdy nemôžu pohnúť doľava, čísla doprava. Napríklad prvok 7 vpravo hore sa posunul doľava, prvok N vpravo dole nadol. Žiaden prvok sa nepohol do smeru, ktorý mu zakazuje jeho typ. Porovnanie stavov automatu je na obrázkoch 6.7 a 6.8.



Obr. 6.7: t = 0



Obr. 6.8: t = 1

Pri pozorovaní tohto efektu vzniká po viacerých iteráciách zaujímavý pohľad. Prvky sa z počiatočného stavu 6.9, kedy sú situované všemožne po automate, začnú postupne zarovnávať na jednu zo strán automatu. V čase t = 4 6.10 sa čísla zbiehajú naľavo,

napravo sú len písmená L a H. V čase t = 19 6.11 je už zarovnanie markantnejšie, ešte stále sa niektoré prvky nachádzajú aj v strednej oblasti automatu. V čase t = 49 6.12 sa už skoro všetky prvky zarovnali na adekvátnu stranu. Prvky ktoré sú ešte stále v strede automatu sa náhodou pohybovali hlavne nahor a nadol, napríklad prvky O a J.

Obr. 6.9: t = 0

Obr. 6.10: t = 4

Obr. 6.11: t = 19

Obr. 6.12: t = 49

6.1.4 Výmena na základe typu

Každý prvok sa vymieňa s iným náhodným prvkom rozdielneho typu. V každej párnej iterácii budú určité bunky obsadené číslami a iné písmenami, v každej nepárnej iterácii sa tieto bunky vystriedajú. Každé písmeno vymení číslo, napríklad prvok L naľavo za prvok 9, naopak každé číslo vymení písmeno, napríklad prvok 6 vpravo hore za prvok Z. Porovnanie stavov je na obrázkoch 6.13 a 6.14.

Obr. 6.13: t = 0

Obr. 6.14: t = 1

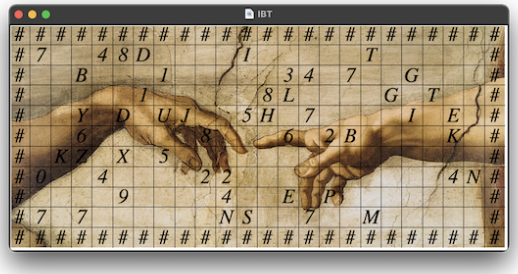
Za použitia náhodnosti nie je vždy úplne zrejmé, ktorý prvok sa vymieňa s ktorým. Prvok U (vľavo) sa vymieňa s prvok 8 (stred), ich poloha je zmenená o náhodnosť. Porovnanie stavov je nižšie na 6.15 a 6.16.

Obr. 6.15: t = 0

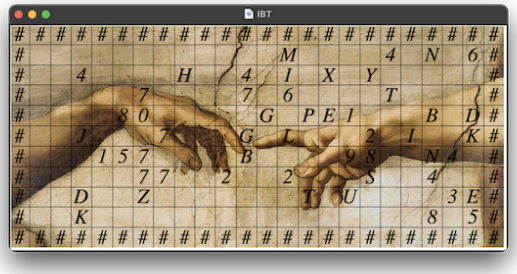
Obr. 6.16: t = 1

6.1.5 Rozptýlenie

Prvky sú v počiatočnom stave náhodne rozhodené po automate. V každom kroku automatu sú prvky opätovne náhodne rozmiestnené. V porovnaní s nasledujúcim stavom sa zmenili ako polohy obsadených buniek, tak aj polohy jednotlivých prvkov, napríklad prvok M sa presunul zdola nahor. Porovnanie stavov je na obrázkoch 6.17 a 6.18.

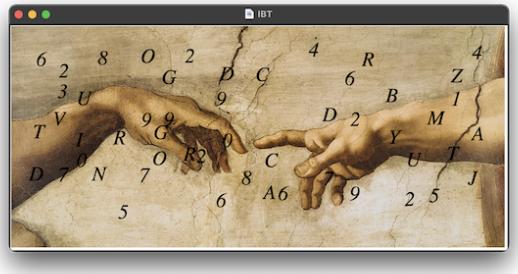


Obr. 6.17: t = 0

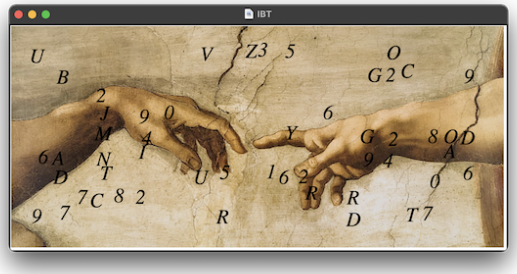


Obr. 6.18: t = 1

Náhodné rozptýlenie v kombinácii s prvkom náhodnosti vytvára zaujímavé rozloženia prvkov. Jednotlivé stavy nejavia súvislosť, ich porovnanie je na obrázkoch 6.19 a 6.20.



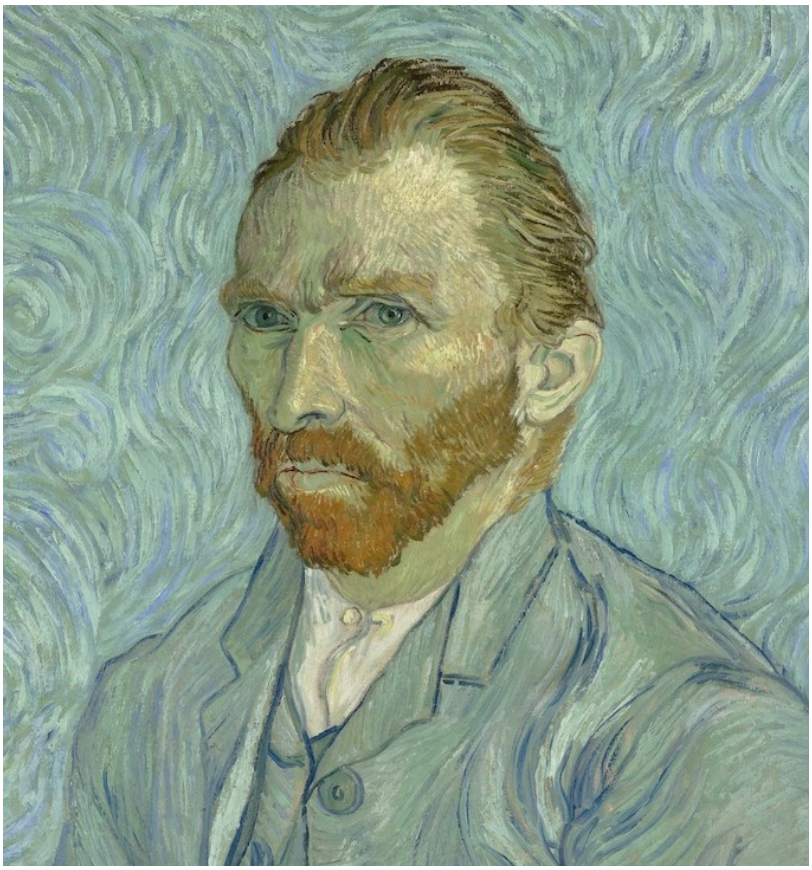
Obr. 6.19: t = 0



Obr. 6.20: t = 1

6.2 Vincent Van Gogh, autoportrét

Druhým obrazom ktorý použijem pre účely testovania bude olejomaľba, *autoportrét Vincenta Van Gogha (1853-1890)* 6.21. Považuje sa za jeho posledný autoportrét. Je zaujímavý tým, že postava je natočená takým spôsobom, že nie je vidno Van Goghovo ucho, ktoré si údajne odrezal. Toto dielo zaraďujeme do umeleckého smeru post-impresionizmus. Obraz má pomerne rovnakú výšku a šírku.



Obr. 6.21: Autoportrét, Vincent Van Gogh, September 1889 [3]

6.2.1 Parametre

Výška obrazu 728 pixelov

Šírka obrazu 681 pixelov

Výška automatu $\frac{728}{30} \doteq 24$ buniek

Pomer šírky a výšky $\frac{681}{728} \doteq 0.93$

Šírka automatu $24 * 0.93 \doteq 22$ buniek

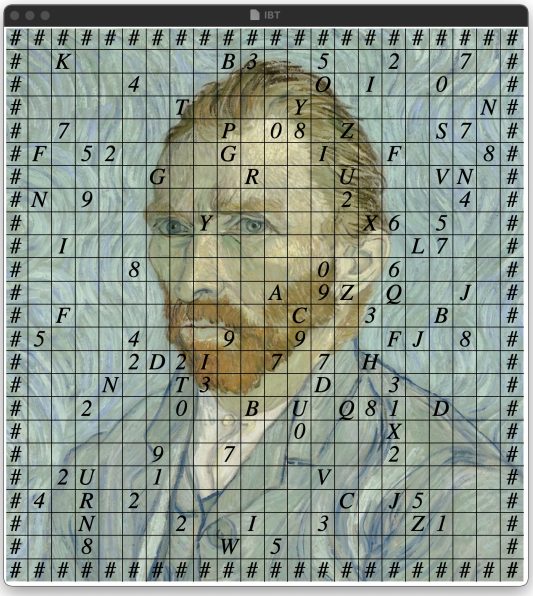
Výška bunky $\frac{728}{24} \doteq 30.33$ pixelov

Šírka bunky $\frac{681}{22} \doteq 30.95$ pixelov

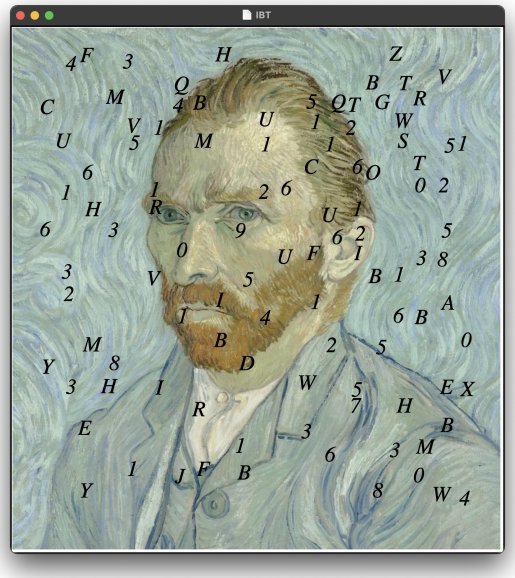
Počet buniek $24 * 22 = 528$ buniek

Počet prvkov jedného typu $528 * 0.1 \doteq 53$ prvkov

Automat ma s hraničnými oblasťami dvadsaťdva buniek na šírku a dvadsaťštyri na výšku, to je takmer rovnaký počet buniek v každom smere. Každá bunka má na výšku a šírku niečo cez tridsať pixelov, rozmer bunky je teda opäť približne tridsať pixelov na výšku a šírku. Automat v každom stave obsahuje päťdesiattri čísiel a päťdesiattri písmen, spolu stošesť prvkov 6.22. Bez pomocných čiar, hraničných prvkov a s prvkom náhodnosti pri vykresľovaní môže vyzerať počiatočný stav aj nasledovne 6.23.



Obr. 6.22: Režim ladenia



Obr. 6.23: Bez režimu ladenia

6.3 Campbellova polievka

Tretí obraz ktorý som sa sem rozhodol zaradiť je obraz plechovky s kuracím vývarom 6.24. Jeho autorom je *Andy Warhol (1928-1987)*, maliar, filmový tvorca a dôležitá osobnosť minulého storočia vrámci hnutia pop art. Medzi jeho ďalšie známe diela patria farebné portréty Merlin Monroe, Micheala Jacksona, či Johna Lenona, tieto diela sú príliš kontrastné a prvky v nich zanikali, preto som som si spomedzi týchto diel vybral práve toto. Výška obrazu je pomerne väčšia ako jeho šírka.

Obr. 6.24: Campbell's Soup Can (Chicken Noodle), 1968 [14]

6.3.1 Parametre

Výška obrazu 761 pixelov

Šírka obrazu 500 pixelov

Výška automatu $\frac{761}{30} \doteq 25$ buniek

Pomer šírky a výšky $\frac{500}{761} \doteq 0.66$

Šírka automatu $25 * 0.66 \doteq 16$ buniek

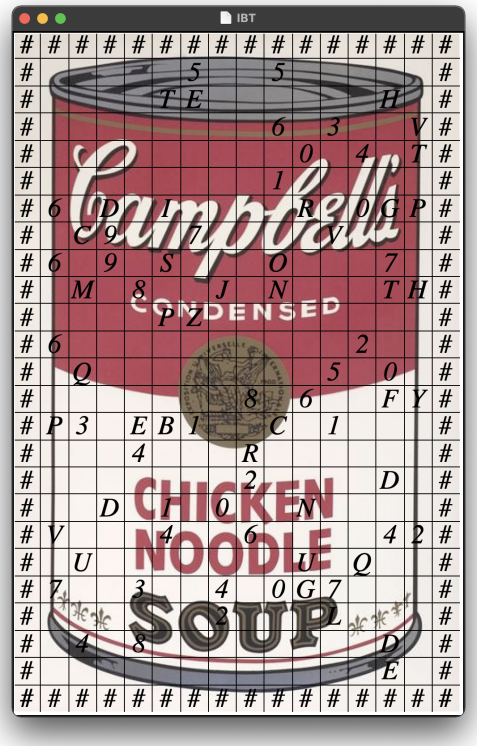
Výška bunky $\frac{731}{25} \doteq 29.24$ pixelov

Šírka bunky $\frac{500}{16} \doteq 31.25$ pixelov

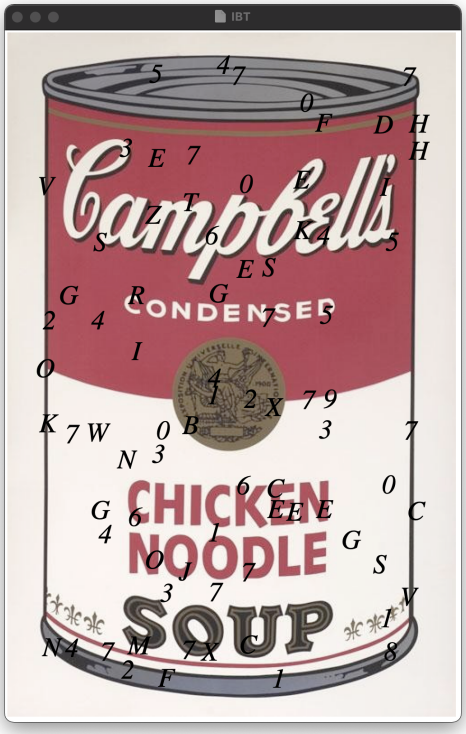
Počet buniek $25 * 16 = 400$ buniek

Počet prvkov jedného typu $400 * 0.1 \doteq 40$ prvkov

Automat má s hraničnými oblasťami dvadsaťpäť buniek na výšku a šestnásť na šírku. Bunky majú opäť približne tridsať pixelov na výšku a šírku. Je v každom stave pokrytý štyridsiatimi písmenami a štyridsiatimi písmenami, spolu osemdesiatymi prvkami 6.25. Počiatočný stav s prvkom náhodnosti môže vyzerať aj takto 6.26.



Obr. 6.25: Režim ladenia



Obr. 6.26: Bez režimu ladenia

Kapitola 7

Záver

Mojim cieľom bolo demonštrovať viacdimenzionálne automaty v umeleckom kontexte, navrhnúť nový typ automatu a skombinovať ho s adekvátnou umeleckou technikou.

Za týmto účelom som preštudoval jednodimenzionálne automaty, teda nedeterministický a deterministický, a jazyky ktoré prijímajú. Ďalej som preštudoval dvojdimenzionálne automaty a známe typy týchto automatov, celulárny, štvorcestný a teselačný. Následne som nachádzal adekvátnu umelecké techniku, identifikoval som, že vhodnou bude koláž, či digitálna koláž.

Na základe týchto znalosti som navrhol nový typ dvojdimenzionálneho automatu, ktorý dokáže ovládať pohyb prvkov koláže. Automat je špecifický tým, že jeho tranzitívna funkcia nie je presne daná, ale je možné ju za behu meniť na základe vstupu užívateľa. Tieto funkcie vytvárajú vizuálne efekty. Pomenoval som ich štvorcestný, štvorcestný na základe typu, výmena na základe typu a rozptýlenie. Každý z nich mení polohy prvkov iným spôsobom.

Špecifikoval som spôsob, akým bude automat rozdelený na moduly, tie som implementoval v jazyku Python3.7.

Automat som otestoval na viacerých dielach rôznych rozmerov a z rôznych umeleckých epoch. Zvolil som renesančný výjav Stvorenie Adama, post impresionistický autoportrét Vincenta van Gogha a Campbellovu polievku, ktorá sa zaraďuje do hnutia pop art. Na každom z týchto obrazov som predviedol, akým spôsobom sú tvorené bunky automatu. Na prvom z nich aj spôsob fungovania jednotlivých efektov a konkrétne stavy, ktoré môžu pri používaní nastať.

Testovaním a experimentovaním som prišiel k dedukcií, že práve tým, že automat mení svoju tranzitívnu funkciu je zaujímavý pre diváka, to je niečo čo nedokážem rozumne zapracovať do tejto práce len vo forme obrázkov.

Pútavé je napríklad, keď spustím štvorcestný efekt na základe typu a po niekoľkých desiatkach iterácií, keď sú prvky už úplne zarovnané na strany, prepnem automat na obyčajný štvorcestný efekt. Vtedy môžem sledovať ako sa oba typy prvkov oddialia od strán automatu, postupne sa dostanú do stredu a začnú sa rôzne medzi sebou premiešavať.

Pekné je tiež sledovať automat, keď prejde štvorcestný efekt na základe typu niekoľko desiatok iterácií a prvky sa zarovnajú na svoje strany. Následne prepnem na výmenu na základe typu a opäť na predchádzajúci efekt. V takom prípade sa prvky dostanú na druhú stranu, než na tú, ku ktorej smerujú kvôli reštrikciám pohybu. Prvky sa začnú posúvať do stredu, obe skupiny sa zrazia, obtrú sa okolo seba a znova sa zarovnajú na svoju stranu.

Ostatný vývoj projektu by sa mohol posúvať rôznymi smermi.

Jedným z nich je preskúmanie automatov vyšších dimenzií. Projekt by mohol pokračovať výskumom trojdimenzionálnych automatov, bolo by potrebné preskúmať

známe automaty o troch dimenziách a nájsť nejakú adekvátnu priestorovú umeleckú techniku, následne vymyslieť nový typ automatu, ktorý by bol schopný pracovať s touto umeleckou technikou a obohatiť ju, implementovať tento automat a nejakým spôsobom aplikovať. Vedel by som si predstaviť trojdimenzionálny priestor tvorený bunkami. Bunky by mohli byť aktívne alebo neaktívne a menili by svoju aktivitu na základe automatu. Ak by sa dostali aktívne bunky do určitej polohy, vytvorili by obraz. Niečo akoby sa hviezdy dostali do určitej konfigurácie a vytvorili by súhvezdie. Toto súhvezdie by sa zobrazilo na rovine, ktorú by tieto hviezdy tvorili.

Ďalej mi napadá preskúmať možnosti efektov s viacerými typmi prvkov než len dvoma. Napríklad by sa mohol rozšíriť štvorcestný efekt tak, že iné prvky by sa nemohli pohnúť nadol, tak by sa zarovnávali nahor.

Napadá mi aj pridať možnosť tvorby vlastných skupín prvkov užívateľovi. V takom prípade by mohlo byť možné ju pomenovať a nahrať sadu obrázkov, ktoré by tvorili túto skupinu prvkov. Ideálne by bolo prepojiť tieto vlastné prvky s efektami, ktoré rátajú s viacerými typmi prvkov.

Samozrejme sa pýta aj možnosť pokračovania v aktuálnom projekte vytváraním nových efektov. Napadá mi efekt posunu doľava alebo doprava, kde by sa prvky posúvali do jeden strany a ak by sa dostali na stranu automatu, v nasledujúcej iterácii by sa objavili na jeho druhej strane.

Za preskúmanie by mohlo stáť aj pohyb prvkov na automate, ktorého bunky nie sú tvaru obdĺžnika, ale iného. Pravdepodobne by sa jednalo o tvary ktoré spolu môžu bezprostredne susediť, čiže napríklad trojuholníkové alebo šesťuholníkové bunky, naopak v prípade päťuholníkových buniek by takéto susedstvo nebolo možné. Automat z takýchto buniek by pravdepodobne nemohol byť vždy obdĺžnikový, ak by sa ale umiestnil tak, že hraničné, neplatné bunky by tvorili hranicu a boli by zrezané, tento rozdiel by zanikol.

Logicky by mohla nasledovať možnosť použiť ako podklad automatu obraz, ktorý nie je obdĺžnikový, ale napríklad tvaru domčeka. Takýto automat by sa prispôsobil tvaru obrazu a generoval by bunky inak než jednoduchým rozdelením na stĺpce a riadky. Najmä niektoré obrazy, ktoré zaraďujeme do moderného umenia sú často atypických tvarov a rozmerov.

Priestor je taktiež v tvorbe efektov, pri ktorých sa mení počet a typ prvkov. V tejto práci sa zaoberám len efektmi, ktoré zanechávajú symetriu v počte prvkov rôznych typov. Takýto asymetrický efekt by mohol meniť typ prvkov v každej iterácii a tým aj ich symbol. Tým sa ale komplikuje prepínanie medzi efektami, pretože niektoré rátajú s rovnakým počtom prvkov.

Ešte by som sa zaoberal efektami, ktoré by prvky usporadúvali do rôznych ornamentov. Premýšľam nad vzormi typickými pre staré slovenské dediny ako Čičmany, taktiež stoja za zmienku orientálne, či rôzne nordické ornamenty. V takomto prípade by bolo asi lepšie, keby prvky neboli písmená a čísla, ale skôr primitívne geometrické tvary, z ktorých by sa tieto vzory mohli rôzne skladať.

Okrem toho je tu priestor pre prácu s farbami prvkov. V tejto práci sa generujú len čierne, no viem si predstaviť, že by po každej iterácii svoju farbu menili, prípadne že by ju prispôsobovali pozadiu aby bol zachovaný aspoň akýsi kontrast, nakoľko čierne môžu v tmavších častiach podkladu zanikať.

Zaujímavé by mohlo byť aj nájsť spôsob, akým by samotný podklad reagoval na polohy prvkov. Ak by napríklad podklad tvoril obrázok váh, mohli by sa tieto váhy preklápať na stranu, na ktorej je viac prvkov. Čím viac by sa na jednej strane obrazu nachádzalo prvkov, tým viac by boli tieto váhy natočené.

Miesto vidím aj v samotných animáciách pohybu prvkov. Aby sa prvky nepresúvali tak, že z jednej bunky zmiznú a v druhej sa objavia, ale aby tento pohyb prestavoval skutočný viditeľný pohyb jednotlivých prvkov. To by však dávalo zmysel len ak by sa prvky presúvali do bezprostredne susedných buniek, v inakšom prípade by na obraze nastal chaos. Ak by sa prvky presúvali, či vymieňali všemožne po automate, viem si predstaviť že by sa buď prepadli alebo rozplynuli a následne vynorili alebo zhmotnili v správnej bunke.

Na um mi tiež prichádzajú prvky viacerých veľkostí. V automate by sa tak nachádzali väčšie prvky, ktoré by zaberali viacero priľahlých buniek. Ak by sa mal takýto prvok posunúť doprava, posunuli by sa tak všetky prvky ktoré tvoria väčší prvok doprava.

Potenciál vidím aj v reakciách prvkov na iné prvky. Prvky by napríklad mohli meniť svoji farbu na základe toho, akým množstvom iných prvkov sú obklopené. Ak by sa pri sebe objavili určité písmená, mohli by vytvoriť jeden veľký prvok, ktorý by predstavoval slovo zložené z týchto písmen. Ak by sa opäť vzdialili, znova by s nimi bolo nakladané ako so samostatnými prvkami.

Vnímam, že najzaujímavejším pokračovaním projektu by bola kombinácia niektorých nápadov, ktoré som spísal, eventuálne úplne iných, ktoré mi vôbec nenapadli. V takom prípade bude ale problémom otázka vstupu užívateľa. Nebude už tak jednoduché prechádzať medzi efektami, preto bude úlohou premyslieť spôsob akým sa bude efekt meniť napríklad v prípade, že sa zmenia počty prvkov rôznych typov a nasledujúci efekt bude počítať s rovnakými počtami. Pravdepodobne bude nutné vytvoriť grafické rozhranie, pretože automat bude obsahovať viacero možností, ktoré budú pre ovládanie cez príkazový riadok obtiažne na uchopenie.

Literatúra

[1] Adamatzky, A. *Game of Life Cellular Automata*. 2010. vyd. Springer, London, 2010. 1-2 s. ISBN 978-1-84996-216-2.

[2] Buonarroti, M. *The creation of Adam* [online]. 1512 [cit. 2021-18-04]. Dostupné z: https://www.michelangelo.org/the-creation-of-adam.jsp.

[3] Gogh, V. V. *Self-portrait* [online]. 1889 [cit. 2021-18-04]. Dostupné z: https://www.vincentvangogh.org/self-portrait.jsp.

[4] Inoue, K. a Nakamura, A. Some properties of two-dimensional on-line tessellation acceptors. *Information Sciences*. 1. vyd. 1977, zv. 13, č. 2, s. 95–121. DOI: https://doi.org/10.1016/0020-0255(77)90023-8. ISSN 0020-0255. Dostupné z: https://www.sciencedirect.com/science/article/pii/0020025577900238.

[5] Kolář, J. *Popisný portrét* [online]. 1963 [cit. 2021-18-04]. Dostupné z: http://www.valentinska.cz/487306-jiri-kolar-pribeh-jiriho-kolare-basnik-noveho-vedeni-kolarovy-nove-metamorfozy-otazka-kolaze.

[6] Kolář, J. *Ruka* [online]. 1964 [cit. 2021-18-04]. Dostupné z: https://www.sypka.cz/ruka-1964/a81/d20578/.

[7] Picasso, P. *A glass and a bottle of Suze* [online]. 1912 [cit. 2021-18-04]. Dostupné z: https://www.kemperartmuseum.wustl.edu/collection/explore/artwork/1105%20%20.

[8] Picasso, P. *Bottle of Vieux Marc,Glass, Guitar and Newspaper* [online]. 1913 [cit. 2021-18-04]. Dostupné z: https://www.tate.org.uk/art/artworks/picasso-bottle-of-vieux-marc-glass-guitar-and-newspaper-t00414.

[9] Průša, D., Mráz, F. a Otto, F. Two-dimensional Sgraffito Automata. *RAIRO - Theoretical Informatics and Applications*. 1. vyd. December 2014, zv. 48, č. 5, s. 6. DOI: 10.1051/ita/2014023.

[10] Rozenberg, G. a Salomaa, A. *Handbook of Formal Languages, Volume 1-3*. 1. vyd. Springer, 1997. 45-49 s. ISBN 3-540-60649-1.

[11] Rozenberg, G. a Salomaa, A. *Handbook of Formal Languages, Volume 1-3*. 1. vyd. Springer, 1997. 49-54 s. ISBN 3-540-60649-1.

[12] Schiff, J. Introduction to Cellular Automata. 2008, s. 79–80.

[13] Smith, T. J. *Two-dimensional automata*. Technical report 2019–637, Queen's University, Kingston, 2019. 5-6 s.

[14] Wahrol, A. *Campbell's Soup Can (Chicken Noodle)* [online]. 1968 [cit. 2021-18-04]. Dostupné z:
https://www.masterworksfineart.com/artists/andy-warhol/campbells-soup.

[15] Ştefan, L. The art of collage and Augmented Reality 2D/3D Techniques. *Aplimat journal*. Február 2012, s. 2–5.

Príloha A

Obsah priloženého pamäťového média

app/ súbory týkajúce sa praktického vypracovania

app/src/ zdrojové kódy

app/images/ obrázky použité pri testovaní

app/requirements.txt zoznam externých knižníc

app/README popis spustenia projektu

doc/ súbory týkajúce sa textu bakalárskej práce

xgazom00.pdf text bakalárskej práce