**Ներածություն**

**Մորֆոլոգիական պատկերի մշակում**

* նախաբան
* էրոզիա
* ընդլայնում
* բացում
* փակում

**Նախաբան**

Մորֆոլոգիանմաթեմատիկան մի լեզու է որը բաղկացած է թեորեմների ցանցից։ Պատկերի մշակման խնդիրների լուծման մեջ մեծ ներդրում ունի մորֆոլոգիական մաթեմատիկան։

Պատկերների արտացոլումը(@reflection) և տրանսլյացիան լայնորեն կիրառվում են ձևաբանության մեջ։

B հավաքածուի արտացոլումը՝ սահմանվում է հետևյալ կերպ՝

= {w | w = -b, for b }

Եթե B-ն իրենից ներկայացնում է 2D պատկեր ապա -ն իրենից կներկայացնի նույն պատկերը սակայն (x,y) կորդինատներին կհամապատասխանեն (-x, -y) կորդինատները։



B-ի տրանսլյացիան z կետի նկատմամբ z = (, ) ` սահմանվում է հետևյալ կերպ՝

= {c | c = b + z, for b }

Եթե B-ն իրենից ներկայացնում է պիքսելների հավաքածու, որը բնորոշում է նկարի միջի օբյեկտը ապա –ը իրենից ներկայացնում է նույն B –ն որտեղ (x,y) կորդինատները փոխարինված են (x+, y+ ) կորդինատներով:

Ներածությունը թե ինչպես ենք էլեմենտների կառուցվածքը ուսումնասիրում մորֆոլոգիաիմեջ բերված է նկարներում ՝



Առաջին տողը իրենից ներկայացնում է էլեմենտների կառուցվածքը ։

Երկրորդ տողը էլեմենտների կառուցվածքը դզևափոխած ուղղանկյունների տեսքով։



Կատարենք գործողությունները A տարրի նկատմամբ օգտագործելով B տարրը։

Տարրերի այն դաշտերը որոնք մուգ են ներկված համարում ենք որ էլեմենտներ կան․

Նոր տարրը ստանալու համար A –ի բոլոր կետերում տեղադրենք B-ն և եթե այդ կետերում պարունակվի B-ն ապա այդ կետերը կհամարենք էլեմենտ։

**Էրոզիա**

Էրոզիան A-ն B-ի վրա ՝ A 󠇝 B սահմանվում է հետևյալ կերպ՝

A 󠇝 B = {z | A }

Բառերով ասված սա նշանակում է որ էրոզիան A-ն B-ի վրա

դա z էլեմենտների բազմություն է , B-ն ենթարկվել է տրանսլյացիաի z –ով, որոնք պարունակվում են A-ում։

Ուրիշ կերպ այս արտահայտությունը կարող ենք գրել հետևյալ կերպ՝

A 󠇝 B = { z | = }

Որտեղ նշանակում է՝

= {w | w A }

Այսինքն այն էլեմենտները որոնք չեն պարունակվում A –ում։

Նկարում պատկերված է օրինակ՝

|  |
| --- |
|  |
| Օգտագործելով էռոզիան փորձում ենք ջնջել նկարի որոշ կոմպոնենտներ` 486 486 թվային նկարը էռոզիային ենք ենթարկում 11 11, 15 15,  45 45 քառակուսի էլեմենտներով։ |

Ընդլայնում

ընդլայնում A-ն B-ի վրա ՝ A 󠇝 B սահմանվում է հետևյալ կերպ՝

A 󠇝 B = { z | A }

Այլ կերպ սա կարող ենք ներկայացնել հետևյալ կերպ՝

A 󠇝 B = {z | A A }

|  |
| --- |
|  |
| Ալգորիթմի օրինակ |

Այս մեթոդը կարող է օգտագործվել տեքստը վերականգնելու համար։

|  |
| --- |
|  |
| Տեքստի վերականգման օրինակ |

**Երկակիություն**

**@duality**

Էրոզիան և ընդլայնումը կապված են իրար հետ։

=

և

=

**Opening and Closing**

@opening A-ն B-ի վրա ՝ A 󠇝 B սահմանվում է հետևյալ կերպ՝

A 󠇝 B = () B

@closing A-ն B-ի վրա ՝ A 󠇝 B սահմանվում է հետևյալ կերպ՝

A 󠇝 B = () B

|  |
| --- |
|  |
| Օրինակ |

Իրենց մեջ ևս գոյություն ունի երկակիություն՝

=

և =

**Որոշ մորֆոլոգիական ալգորիթմների օրինակներ**

Եզրագծերի հայտնաբերումը(Boundary Extraction) A-ն ՝ (A) սահմանվում է հետևյալ կերպ՝

(A) =

|  |
| --- |
|  |
| Նկարում 1-ը նշված է սպիտակ իսկ 0-ն սև |

**Segmentation**

Համակարգչային տեսողության կարևոր խնդիրներից մեկն է հանդիսանում ինֆորմացիայի դուրսբերումը պատկերից։ Որպիսզի կարողանանք մշակել նկարները պետք է այն բաժանենք որոշակի օբյեկտների։

Պատկերների սեգմենտացիան օգտագործվում է թվային պատկերները որոշակի սեգմենտների(մասերի)

բաժանելու համար, այլ կերպ դա նաև անվանում են պիքսելների ցանց կամ սուպեր֊պիքսելենրի հավաքածու։ Պատկերները սեգմենտավորելու իմաստը կայանում է նրանում որ սեգմենտների հետ աշխատանքը զգալիորեն հեշտացնում է պատկերի հետ աշխատանքը ավելի հեշտ է լինում պատկերների մեջ օբյեկտների հետազոտությունը և փոփոխությունը։

Սովորաբար օգտագործվում է պատկերներում օբյեկտների հայտնաբերման և սահմանների հայտնաբերման համար։ Գոյություն ունեն սեգմենտացիայի բազմաթիվ ալգորիթմներ, սակայն խնդրից կախված ալգորիթմները կարող են փոփոխվել, ասինքն մեկ խնդրի համար տրված ալգորիթմը լինի լավը իսկ մյուսի խնդրի համար վատ։

Սեգմենտավորման ալգորիթմների հիմնական դասակարգումներն են՝

1. (Thresholding)Շեմային ալգորիթմները

2. (Алгоритм наращивания областей)(Методы разрастания областей)միջակայքերի աճեցում

3․Սահմանային ալգորիթմները

4. Կլաստերիզացիայի հիմման վրա

**Thresholding**

Պատկերների սեգմենտավորման պարզագույն մեթոդներից մեկը դա շեմային մեթոդն է։

Եթե անհրաժեշտ է ուսումնասիրել նկարը, որի մեջ կան գույների զգալի տատանումներ ապա այդ նկարների համար նպատակահարմար է օգտագործել այս մեթոդը։

Ենթադրենք որ հիստոգրամայի ինտենսիվությունը նկարում համապատասխանում է f(x,y)֊ին, որը իրենից ներկայացնում է լուսավոր օբյեկտները մուգ ֆոնի վրա, այնպես որ պատկերի միջի օբյեկտները և նրանց հետևի ֆոների պիքսելների ինտենսիվության արժեքները խմբավորված են երկու գերիշխող ռեժիմներում նկ10(ա).

|  |
| --- |
|  |
| Նկ10 |

Ակնհայտ եղանակներից մեկը օբյեկտը հետևի ֆոնից առանձնացնելու դա շեմի ընտրությունն է ՝ T որի միջոցով կառանձնացնեք օբյետկը, այսպիսով յուրաքանչյուր (x,y) կորդինատի վրայով կանցնենք և եթե f(x,y) > T կանվանենք օբյեկտի կետ, իսկ մյուս դեպքում կանվանենք հետևի ֆոնի կետ։

Այսպիսով սեգմենտավորված նկարի կտրվի հետևյալ բանաձևով ՝ g(x,y)

|  |  |
| --- | --- |
| g(x,y) = | Եթե f(x,y) T |
| Եթե f(x,y) T |

T-ի առժեքը ամբողջ նկարի համար կառող է լինել նույնը սակայն կախված հարևան պիքսելենրի միջին ինտենսիվությունից կարող է փոփոխվել։

Նկ 10(բ) –ում ցույց է տրված ավելի դժվար շեմավորման պրոբլեմ, որտեղ պիքսելների ինտենսիվության արժեքները խմբավորված են երեք գերիշխող ռեժիմներում , օրինակի համար 2 տիպի սպիտակ օբյեկտները մուգ հետևի ֆոնի վրա։

Այսպիսի իրավիճակում պետք է կատարենք բազմակի շերտավորում, եթե մեր f(x,y) < արժեքից ապա այն պատկանում է 1 օբյեկտին եթե < f(x,y) ապա այն պատկանում է 2րդ օբյեկտին, մնացած այլ օբյեկտներին կամ հետևի ֆոնին f(x,y) > ։

|  |  |
| --- | --- |
| g(x,y) = | Եթե f(x,y) |
| Եթե < f(x,y) |
| Եթե f(x,y) < |

Կլաստերիզացիայի հիմման վրա

K-means ալգորիթմը իրենից ներկայացնում է իտերացիոն ալգորիթմ, որը օգտագործվում է պատկերը մասնատելու համար ՝ K կլաստեռների։

K-means ալգորիթմը կլաստերիզացիայի հայտնի ալգորիթմներից մեկն է հանդիսանում։ Հորինվել է 1950 ական թվականներին։

​**Flood filling**

Լցոնում

Պատկերը սեգմենտավորելու կարևոր ալգորիթմներից մեկն է հանդիսանում flood fill-ը, որը հայտնի է նաև region growing անունով։

**Image pyramids**

**Պատկերի բուրգեր**

Սովորաբար մենք խնդիր ենք ունենում օրիգինալ նկարը չափերը փոփոխելու, դրա համար գոյություն ունեն 2 մեթոդնոր՝

* Մեծացնել չափերը(մոտիկացնել)
* Փոքրացնել չափերը (հեռվացնել)

Պատկերի բուրգերը իրենցից ներկայացնում են պատկերների հավաքածու որոնք ստացվում են օրիգինալ նկարից, օրիգինալ նկարը դաունսեմպլինգի(downsampling) ենթարկելով մինչև մի ցանկալի տարանցիկ կետի հասնելը։

Դաունսեմպլինգը գործողություն է որի ընթացքում փոխում ենք նկարին տրված չափերը 72 dpi. : Սա կարող է զգալիորեն փոքրացնել նկարի ծավալը։

Դաունսեմպլինգի վատ կողմը կայանում է նրանում, որ երբ նկարը մոտիկացնում ենք 100%֊-ից ավել ապա արդեն նկարը սկսում է երևալ ոչ ճշգրիտ։

Այսպիսով յուրաքանչյուր նկար դառնում է ¼ մասը իր նախորդի։ Այս ալգորիթմը սովորաբար օգտագործվում է պատկերների սեգմենտացիայի մեջ, փոքրացնում է նկարի չափերը որից հետո հնարավոր է լինում որոշակի գործողություններ կատարել պատկերի հետ և հետո ավելի բարձր որակով նկարը ետ բերել։

Պատկերի բուրգերի հիմնկան տեսակները ՝

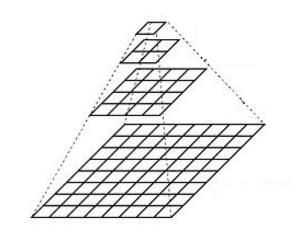
* Գաուսիան բուրգեր
* Լապլասիան բուրգեր

**Գաուսիան բուրգեր ։**  Օգտագործվում է պատկերները դաունսեմպլինգի ենթարկելու համար։

**Լապլասիան բուրգեր**։ Օգտագործվում է upsampled-ի ենթարկելու համար, որի ժամանակ բուրգի պատկերների փոքր չափերի նկարների չափերը կարելի է մեծացնել։

**Գաուսիան բուրգեր**

* Պատկերացնենք մի բուրգ որի կազմված է բազմաթիվ շերտերից և որքան ավելի վերև ենք բարձրանում այնքան այդ շերտերի չափերը փոքրանում են։



* Շերտերը համարակալենք ներքևից վերև, այնպես որ լինի ավելի փոքր քան -ը։
* Ամեն (i+1)-րդ շերտը ստանալու համար մենք պետք է անենք հետևալ գործողությունները:

1. Կոնվոլուցիայի(Convolve) ենթարկենք -ը Գաուսիան միջուկի(kernel) հետ։

* Մենք հեշտությամբ կարող ենք համոզվել նրանում որ ելքային պատկերը ստացվում է մուտքային պատկերի ¼ տարածքի չափով։

|  |  |
| --- | --- |
|  |  |
| Օրիգինալ մուտքային պատկերը | Ելքային պատկերը դաունսեմպլիգի ենթարկելուց հետո |

* Մենք նաև կարող ենք իրականացնել հակառակ գործողությունը՝ այսինքն 4 անգամ մեծացնել(upsampling):

|  |  |
| --- | --- |
|  |  |
| Օրիգինալ մուտքային պատկերը | Ելքային պատկերը upsampling-ի ենթարկելուց հետո |

Եթե փորձենք պատկերը մեկ անգամ ենթարկել դաունսամպլինգի մյուս անգամ upsamplingi կստանանք հետևյալ պատկերը՝

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Օրիգինալ նկար | Դաունսամպլինգի ենթարկելուց հետո | Upsampling –ից հետո |

**Կոնվոլուցիա։**

Պատկերների մշակման մեջ միջուկը(kernel), կոնվոլյուցիայի մատրիցան կամ դիմակը(mask) իրենցից ներկայացնում են փոքր չափերի մատրիցա։ Դա շատ օգտակար է քողարկման(bluring), եզրերի հայտնաբերման և այլ շատ վայրերում։

Միջուկը օրիգինալ պատկերի հետ կոնվոլուցիայի են ենթարկում և կախված միջուկի մատրիցայի չափից և կառուցվածքից կարող են ստացվել տարբեր արդյունքներ։

|  |  |  |
| --- | --- | --- |
| օպերացիան | միջուկը | պատկերի արդյունքը |
| Ինքնության ճանաչում |  | Vd-Orig.png |
| Եզրերի հայտնաբերման համար |  | Vd-Edge1.png |
|  | Vd-Edge2.png |
|  | Vd-Edge3.png |
| Սրում |  | Vd-Sharp.png |
| Տուփի քողարկում  Box blure |  | Vd-Blur2.png |
| Գաուսիան քողարկում 3x3 |  | Vd-Blur1.png |
| Գաուսիան քողարկում 5x5 |  | Vd-Blur Gaussian 5x5.png |
| [**Unsharp masking**](https://en.wikipedia.org/wiki/Unsharp_masking) Շրջանային դիմակավորում  5 x 5 |  | Vd-Unsharp 5x5.png |

Մատրիցների կոնվոլուցիայի պարզագույն օրինակ կարող ենք բերել հետևյալը՝

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | 1 | 2 | 3 | | 4 | 5 | 6 | | 7 | 8 | 9 | | Օրիգինալ | | | | |  |  |  | | --- | --- | --- | | -1 | -2 | -1 | | 0 | 0 | 0 | | 1 | 2 | 1 | | Միջուկ | | | | |  |  |  | | --- | --- | --- | | -13 | -20 | -17 | | -18 | -24 | -18 | | 13 | 20 | 17 | | Ելքային | | | |

Պատկերը բաժանում ենք փոքր մատրիցաների որոնց չափերը կարող են համընկնել կամ չհամընկնել միջուկի չափերի հետ։

Առաջին քայլով կատարում ենք միջուկի պտույտ 180-ով,

որից հետո ստանում ենք հետևյալ մատրիցը

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |
| Միջուկ | | |

Այնուհետև կատարում ենք մատրիցաների բազմապատկում։ Միջուկից դուրս ընկած կետերը համարում ենք 0-ներ։

|  |  |
| --- | --- |
| y[0,0] | y[0,0] = 0\*1+0\*2+(-2\*4) + (-1\*5) = -13 |
| y[1,0] | y[1,0] = 0\*1 + 0\*2 + 0\*3 + (-1\*4) + (-2 \* 5) \* (-1 \* 6 ) = -20 |
| y[2,0] | y[2,0] = 0\*2 + 0\*3 + (-1 \* 5 ) + (-2 \* 6) = -17 |
| y[0,1] | y[0,1] = 2\*1+1\*2+0\*4+0\*5+(-2\*7) + (-1\*8) = -18 |
| y[1,1] | y[1,1] = 1\*1 + 2\*2 + 1\*3 + 0\*4 + 0\*5 + 0\*6 + (-1\*7) + (-2 \* 8) + (-1 \* 9) = -24 |
| y[2,1] | y[2,1] = 1\*2+2\*3+0\*5+0\*6+(-1\*8) + (-2\*9) = -18 |
| y[0,2] | y[0,2] = 2\*4+1\*5+0\*7+0\*8 = 13 |
| y[1,2] | y[1,2] = 1\*4+2\*5+1\*6+0\*7+0\*8+0\*9 = 20 |
| y[2,2] | y[2,2] = 1\*5 + 2\*6 + 0\*8 + 0\*9 = 17 |

**Լապլասիան բուրգեր**

* Լապլասիան բուրգը շատ նման է եզրագծերի հայտնաբերման ալգորիթմին
* Բուրգերի պիքսելների մեծ մասը 0 են(սև)
* Այս ալգորիթմը շատ է օգտագործվում պատկերներիսեղղման մեջ։

Լապլասիան բուրգերը ստացվում են Գաուսիան բուրգերի հիմման վրա՝

= – EXPAND[]

= – EXPAND[]

= – EXPAND[]

=

……………………

Այսպիսով մենք ստացանք Լապլասիան բուրգերը Գաուսիանի հիմման վրա, սակայն հնարավոր է և հակառակ գործողությունը։

=

= + EXPAND[]

= + EXPAND[]

= + EXPAND[]

-ը մեզ մոտ կլինի օրիգինալ պատկերը։

Խնդրի դրվածքը

Խնդիր է դրծավ ուսումնասիրել պատկերների հետ աշխատանքը, որը հանդիսանում է արհեստական ինտելեկտի ճյուղերից մեկը, և հետազոտել օբյեկտները պատկերների մեջ։

Հիմնավորում

Այդ ուսումնասիրությունները մեծ կարևորություն ունեն մարդկանց համար, օրնակ տեսախցիկները որոնք վերահսկում են երթևեկությունը աշխատում են այդ ուսումնասիրությունների հիմման վրա, կամ դեմքի հայտնաբերումը նկարի մեջ։