Universiteit Gent
Faculteit Wetenschappen

Vakgroep
Toegepaste wiskunde en informatica
Voorzitter: Prof. G. VANDEN BERGHE

EVALUATIE VAN VAAGLOGISCH GEBASEERDE FILTERS VOOR RUISONDERDRUKKING

door
Björn BLANCKAERT

Promotor: Prof. Dr. E. E. Kerre
Scriptiebegeleiders:
Dr. M. NACHTEGAEL
D. VAN DER WEKEN

Scriptie ingediend tot het behalen van de academische graad van licentiaat informatica

Academiejaar 2003-2004

Voorwoord

De tijd vliegt snel, gebruikt hem wel... Dit jaar heb ik aan de lijve ondervonden dat deze raad geen praat bij de vaak is. Maar ik durf zeggen dat mijn tijd goed besteed is. Wanneer nu analoge-radio-werpen kanshebber is om een nieuwe olympische discipline te worden, zal binnenkort waarschijnlijk hetzelfde gebeuren met ons tv-toestel. Het digitale tijdperk is al een tijdje bezig, maar begint nu vaste grond onder de voeten te krijgen. De kijker zal veel interactiever worden en het beeld zal veel beter worden, waardoor ook de kijker beter zal worden. Het is dan ook niet verwonderlijk dat beeldverwerkingstechnieken een hot item zijn, waaraan ik met plezier m'n (kiezel)steentje bijgedragen heb.

Een voorwoord is ook traditiegetrouw een moment om even stil te staan en enkele mensen te bedanken. Dit doe ik dan ook van ganser harte. Eerst en vooral wil ik mijn thesisbegeleiders Dr. Mike Nachtegael en Dietrich Van der Weken bedanken. Zonder hun steun en stimulatie zou ik hier volgend jaar misschien nog steeds zitten. Ook Prof. Dr. E. E. Kerre verdient een woord van dank voor de uiterst nauwkeurige correcties. Ik mag ook niet nalaten mijn ouders te bedanken. Zonder hen zou ik hier überhaupt niet zitten en hun financiële en logistieke steun was onontbeerlijk. Mijn vrienden zorgden voor de o zo noodzakelijke ontspanning. Ik wil in het bijzonder Nikcy Van Thuyne bedanken, die naast de ontspanning ook zorgde voor enkele correcties. Als laatste zou ik ook nog mijn vriendin willen bedanken voor haar motivatie in tijden waar het niet zo goed ging. En als allerlaatste nog een dankwoord voor Prof. Per-Olav Rusås en zijn studenten van het Noorse Østfold University college voor het ontwikkelen en beschikbaar stellen van een eenvoudige imageprocessing-toolkit voor java.

EVALUATIE VAN VAAGLOGISCH GEBASEERDE FILTER VOOR RUISONDERDRUKKING

door
Björn BLANCKAERT

Scriptie ingediend tot het behalen van de academische graad van licentiaat informatica

Academiejaar 2003-2004

Promotor: Prof. Dr. E. E. KERRE
Begeleiders: Dr. M. NACHTEGAEL en D. VAN DER WEKEN
Faculteit Wetenschappen
Universiteit Gent

Vakgroep Toegepaste Wiskunde en Informatica
Voorzitter: Prof. G. VANDEN BERGHE

Samenvatting

Er werden al verscheidene vaaglogisch gebaseerde filters voor ruisonderdrukking ontwikkeld. Deze werden meestal vergeleken met filters die minder goed presteerden. In deze scriptie worden de filters besproken en geïmplementeerd in een JAVA-programma. Daarna wordt gebruik gemaakt van de implementatie om de filters te evalueren.

Trefwoorden: filters, ruisonderdrukking, vaaglogica, vergelijking

Inhoudsopgave

Hoofdstuk 1

Inleiding

1.1 Functie van beelden

De mens is altijd al gefascineerd geweest door beelden. Vanaf het prille begin begon men de wereld na te tekenen op een rotswand of in een boom. Toen werd het beeld nog gebruikt om gunsten af te smeken van de goden. Langzamerhand lukte het om de wereld steeds realistischer weer te geven. Schilderen werd een kunst. Beelden werden gebruikt om te pronken met rijkdom of opdat het nageslacht zou kunnen voorstellen hoe het leven in die tijd was of had moeten zijn. Deze laatste functie werd een hondertal jaren geleden overgenomen door het fototoestel, dat ons een bijna perfecte voorstelling van de echte wereld aanbiedt.
Hedentendage worden beelden vooral gebruikt als informatiedragers. Een beeld zegt meer dan duizend woorden is niet voor niets een graag geciteerde uitspraak. Deze functie als informatiedrager heeft tal van toepassingen in het dagelijks leven. Zo zijn beelden fundamenteel voor de meeste reclametoepassingen, kan het ontwerp van een nieuwe machine moeilijk uitgelegd worden zonder tekening, worden foto's van een gebied gebruikt om een militaire strategie te bepalen en zo kan nog een hele resem andere voorbeelden gegeven worden.

1.2 Modellering van beelden

Om de wereld te kunnen voorstellen maken we gebruik van een functie die de wereld(W) afbeeldt op een representatie ervan(S). Het is niet mogelijk om aan de hand van de inverse functie de wereld te zien, aangezien veel dingen verborgen blijven, niet zichtbaar zijn. Dit is een probleem wanneer we de wereld willen voorstellen binnen een computer, maar aangezien deze voorstelling steeds een doel heeft, is het niet nodig de werkelijkheid exact af te beelden, maar een benadering ervan die leidt tot het bereiken van ons doel. In deze tekst is ons doel het manipuleren van een beeld.
De gebruikte notaties kunnen teruggevonden worden in bijlage A, maar de belangrijkste zal ik hier nog eens vermelden.

- (i, j) =positie van een pixel.
- $A(i, j)$ =grijswaarde van de pixel op positie (i, j).
- A =het beeld
- L =de maximum grijswaarde
- A' =het gefilterd beeld

Om een beeld digitaal te kunnen bewerken, moet het beeld ook digitaal kunnen opgeslagen worden. Er moet een manier zijn om het beeld voor te stellen binnen een computer en hiervoor moet een analoog signaal, met name weerkaatsing van licht, omgezet worden naar een digitaal signaal. Dit wordt wiskundig gemodelleerd aan de hand van een functie f, die een plaats afbeeldt op een lichtsterkte of energie. In grijswaardebeelden is $f(\mathbf{x})$ een scalair, in kleurenbeelden een 3-dimensionele vector die de RGB[1]-waarden van dat punt bevat.
Op de computer worden de functiewaarden van f bijgehouden in een $m \times n$ matrix A, waarbij $A(i, j) = f(i, j)$. De waarden van $f(i, j)$ zijn scalairen aangezien we hier uitsluitend werken met grijswaardebeelden. Ieder element van de matrix wordt een pixel[2] genoemd. De waarden die $f(i, j)$ kan aannemen liggen in het interval $[0, L]$. Meestal is $L = 256$, want per slot van rekening werken we nog altijd met de computer waarop alles in bytes wordt opgeslagen. Op deze manier bekomt men een matrix met waarden die dan aan het beeldscherm doorgegeven kunnen worden om lichtintensiteiten te simuleren en beelden weer te geven.

[1] rood-groen-blauw

[2] de kleinste eenheid op een scherm of op een printer waarvan de kleur of helderheid digitaal vastgelegd kan worden

1.3 Ruis

Eenvoudig gezegd is ruis een afwijking van het beeld zoals het oorspronkelijk bedoeld was. Ruis komt dan naar voor als lichte of donkere vlekken op het beeld en kan het beeld als heel onnatuurlijk laten voorkomen. Verschillende factoren veroorzaken ruis, zoals de aanwezigheid van licht op het moment dat de foto genomen werd, de sluitertijd, de sensortemperatuur van een digitaal fototoestel, stofdeeltjes op de lens, ...
De drie meest voorkomende ruistypes worden hieronder gedefinieerd.

1.3.1 Zout&peper-ruis

Bij zout&peper-ruis zijn er drie mogelijkheden:

- de pixelwaarde blijft ongewijzigd;
- de pixelwaarde neemt de maximumwaarde L aan;
- de pixelwaarde neemt de minimumwaarde 0 aan.

Op deze wijze worden er zwarte en witte korrels gevormd op het beeld. Zout&peper-ruis wordt bepaald door de ruisdichtheidsfactor $\delta \in [0, 1]$ die de verhouding van de vervuilde pixels t.o.v. het totale aantal pixels aangeeft. In een $m \times n$-beeld zijn bijgevolg $\delta \times (m \times n)$-pixels vervuild. Zout&peper-ruis is een bijzonder geval van impulsruis[3]. Zout&peper-ruis is het ruistype waarvoor het meest aantal filters ontwikkeld werden, dit is namelijk het ruistype dat het eenvoudigst te verwijderen is. Een voorbeeld van zout&peper-ruis vind je in figuur 1.1.

1.3.2 Gaussische ruis

Gaussische ruis wordt ook wel additieve ruis genoemd, omdat het vervuild beeld kan opgevat worden als de som van het originele $m \times n$-beeld en een $m \times n$-matrix met random-waarden met een normale verdeling, gemiddelde g(meestal 0) en variantie σ^2. Hoe groter σ, hoe sterker de ruis. Met behulp van statistiek, kunnen geschikte filters ontworpen worden voor dit type ruis. Een voorbeeld van gaussische ruis vind je in figuur 1.2.

[3]De vervuilde pixels nemen in plaats van de maximum- en minimumwaarden, waarden aan die opmerkelijk hoger of lager liggen dan de waarden van de omliggende pixels.

1.3.3 Speckle ruis

Speckle ruis wordt ook wel multiplicatieve ruis genoemd, omdat het ruisbeeld multiplicatief bekomen wordt uit het origineel beeld:

$$\begin{aligned} A'(i,j) &= A(i,j) + N(i,j) \cdot A(i,j) \\ &= (1 + N(i,j)) \cdot A(i,j), \end{aligned}$$

met $N(i,j)$ een $m \times n$-matrix met random-waarden met een normale verdeling, gemiddelde g en variantie σ^2. Hoe groter de variantie, hoe sterker de ruis. Dit type ruis komt verder niet meer aan bod, maar wordt hier voor de volledigheid toch vermeld.

Figuur 1.1: Zout&peper-ruis: (a)Origineel beeld; (b)$\delta = 0.05$; (c)$\delta = 0.25$; (d)$\delta = 0.50$.

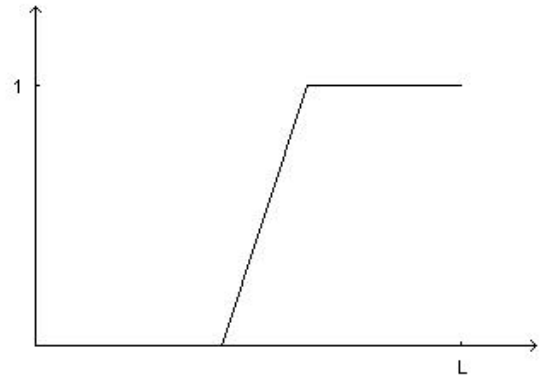
Figuur 1.2: Gaussische ruis: (a)Origineel beeld; (b)$\sigma = 7$; (c)$\sigma = 15$; (d)$\sigma = 25$.

1.4 Vaagverzamelingen en vaaglogica

Vaagverzamelingenleer en vaaglogica werden voor het eerst vermeld in het midden van de jaren zestig door prof. Lotfi A. Zadeh (university of California) en is één van de meest ontwikkelde modellen om vaagheid en onzekerheid te modelleren. In tegenstelling tot de klassieke benadering dat een object tot een zekere verzameling behoort, suggereert de vaagverzamelingenleer dat het behoren van een element tot een bepaalde verzameling ook gradueel kan overgaan van niet-lidmaatschap tot volledig lid. Deze technieken hebben hun nut al bewezen op verscheidene wetenschappelijke vlakken en in toepassingen uit het dagelijks leven. Zo bestaat er reeds een vaaglogische wasmachine.
Wat is nu een vaagverzameling? Zij X een universum en P een eigenschap. Dan is een vaagverzameling A_P in X een $X - [0, 1]$ afbeelding waarvoor voor alle x in X $A_P(x)$ de graad voorstelt waarin x aan de eigenschap P voldoet. Wanneer $A_P(x) = 1$, dan betekent dit dat het element x perfect aan P voldoet, terwijl $A_P(x) = 0$ betekent dat x absoluut niet aan P voldoet. De functie waardoor A_P kan voorgesteld worden, wordt de lidmaatschapsfunctie genoemd.
Vaaglogica is de uitbreiding van de klassieke logica die werkt op de verzameling $\{0, 1\}$ naar het volledige interval $[0, 1]$. De operatoren zoals ze bestaan in de klassieke logica, worden uitgebreid naar de vaaglogica. Zo kan bijvoorbeeld de conjunctie $\wedge$ uitgebreid worden naar de min-operator.
Wat kunnen we nu doen met deze informatie? Net zoals in de klassieke logica, kunnen we regels opstellen. Bijvoorbeeld:

ALS (de pixel heeft een `hoge grijswaarde`)
OF (de pixel heeft een `lage grijswaarde`)
DAN de pixel is `waarschijnlijk` ruis

Aangezien de linguïstische termen `hoge grijswaarde` en `lage grijswaarde` een imprecieze omschrijving zijn, is dit een regel uit de vaaglogica. Deze termen worden gemodelleerd door middel van een lidmaatschapsfunctie. Een voorbeeld van de lidmaatschapsfunctie van een vaagverzameling `hoge grijswaarde` wordt getoond in figuur 1.3. Verschillende types van functies worden gebruikt in de besproken filters.



Figuur 1.3: Voorbeeld van de lidmaatschapsfunctie van een vaagverzameling 'hoge grijswaarde'.

Hoofdstuk 2

Overzicht van filters voor ruisonderdrukking

2.1 Klassieke filters

2.1.1 Mediaanfilter - MF

De mediaanfilter werkt met een $r \times r$ invoervenster (met $r = 2l + 1$ en l een natuurlijk getal, r moet dus oneven zijn). Dit is een matrix van grootte $r \times r$ met de te filteren pixel als centrale pixel. Iedere pixel van het beeld wordt vervangen door de mediaan van het invoervenster met deze pixel als centrale pixel. Op deze manier worden extreme grijswaarden geëlimineerd en wordt impulsruis verwijderd. Het gefilterd beeld is wel waziger dan het oorspronkelijke beeld en fijne details worden verwijderd. Symbolisch:

$$A'(i,j) = \text{med}(W_{(i,j)}),$$

hierbij is $W_{(i,j)}$ het invoervenster met centrale pixel (i,j). Een beeld gefilterd met de mediaanfilter wordt getoond in figuur 2.1.

2.1.2 Gemiddelde waarde filter - MAV

De moving average filter werkt eveneens met een $r \times r$ invoervenster. Bij deze filter wordt iedere pixel in het beeld vervangen door de gemiddelde waarde van het invoervenster. Dit is een geschikte filter voor het verwijderen van gaussische ruis. Voor impuls-ruis daarentegen is deze filter absoluut niet geschikt, aangezien donkere en lichte vlekken zorgen voor het omlaag of omhoog jagen van het gemiddelde. Een beeld gefilterd met de moving average filter wordt getoond in figuur 2.2.

Figuur 2.1: Mediaanfilter met venstergrootte 3×3 toegepast op Lena vervuild met zout&peper-ruis: (a)Origineel beeld; (b)$\delta = 0.05$; (c)$\delta = 0.25$; (d)$\delta = 0.50$.

Figuur 2.2: Gemiddelde waarde filter met venstergrootte 3×3 toegepast op Lena vervuild met gaussische ruis: (a)Origineel beeld; (b)$\sigma = 7$; (c)$\sigma = 15$; (d)$\sigma = 25$.

2.1.3 Meerstaps mediaanfilter - MMF

De multilevel median filter werd ontworpen om impuls ruis te verwijderen en werkt met een $r \times r$ venster met de te filteren pixel als centrale pixel. In tegenstelling tot de gewone mediaanfilter, waar de mediaan van het invoervenster wordt berekend, wordt hier voor elk van de 4 richtingen horizontaal, verticaal en de 2 diagonalen de mediaan berekend om zo in verschillende fasen de uiteindelijke waarde te bepalen.
Definieer vier deelvensters van het oorspronkelijke $(2l+1) \times (2l+1)$ invoervenster $W_{(i,j)}$ als volgt:

$$\begin{aligned} W_{(i,j),1} &= \{A(i, j+k) : -l \leq k \leq l\} \\ W_{(i,j),2} &= \{A(i+k, j+k) : -l \leq k \leq l\} \\ W_{(i,j),3} &= \{A(i+k, j) : -l \leq k \leq l\} \\ W_{(i,j),4} &= \{A(i+k, j-k) : -l \leq k \leq l\}. \end{aligned}$$

Stel nu $\text{med}_x(i,j)$, voor $x = 1, 2, 3, 4$ de mediaan van ieder deelvenster. Dan kunnen 2 variabelen gedefinieerd worden:

$$\begin{aligned} \text{med}_{max}(i,j) &= \max_{x=1}^{4} \text{med}_x(i,j) \\ \text{med}_{min}(i,j) &= \min_{x=1}^{4} \text{med}_x(i,j). \end{aligned}$$

Dan wordt de output van de multilevel mediaan filter:

$$A'(i,j) = \text{med}(\text{med}_{max}(i,j), \text{med}_{min}(i,j), A(i,j))$$

Een beeld gefilterd met de multilevel medianfilter wordt getoond in figuur 2.3. Door de typische vorm van de deelvensters, komen streepvormige structuren voor.

2.1.4 Gaussische filter

De gaussische filter werd ontwikkeld om gaussische ruis te verzachten. Hij vervangt iedere pixel van het beeld door een lineaire combinatie van de grijswaarden in een 3×3 omgeving van die pixel, ongeacht de positie ervan:

$$A'(i,j) = \sum_{m=-r}^{r} \sum_{n=-r}^{r} w(m,n) \cdot A(i-m, j-n).$$

Bij de gaussische filter is

$$w(m,n) = \frac{w_0(m,n)}{\sum_{m,n=-1}^{1} w_0(m,n)},$$

Figuur 2.3: Meerstaps mediaanfilter met venstergrootte 7×7 toegepast op Lena vervuild met zout&peper-ruis: (a)Origineel beeld; (b)$\delta = 0.05$; (c)$\delta = 0.25$; (d)$\delta = 0.50$.

met:

$$w_0(m,n) = e^{-\frac{k^2+l^2}{2\sigma^2}}.$$

Hierbij is σ een nader te specifiëren parameter. Wanneer de filter effectief gebruikt wordt voor gaussische ruis, dan neemt men voor σ de standaardafwijking van de gaussische ruis, waarbij de maximale grijswaarde L gelijk is aan 1. Bij een andere maximale grijswaarde L, kan men σ als volgt bepalen:

$$\sigma = (\frac{\sigma_L}{L})^2$$

Een beeld gefilterd met de moving average filter wordt getoond in figuur 2.4.

Figuur 2.4: Gaussische filter toegepast op Lena vervuild met gaussische ruis: (a)Origineel beeld; (b)$\sigma = 7$; (c)$\sigma = 15$; (d)$\sigma = 25$.

2.2 Vaag-klassieke filters

2.2.1 Vaaglogische mediaanfilter - FMF

Idee Het probleem met de klassieke mediaanfilter is dat elke pixel verwerkt wordt en niet enkel de geïnfecteerde pixels. Dit maakt het gefilterde beeld wazig. Daarom maakt de FMF-filter[1] gebruik van vaagregels om te bepalen in welke mate een pixel geïnfecteerd is, waarna deze informatie gebruikt wordt om een gewicht toe te kennen aan de uitvoer van de mediaanfilter. Deze filter werd ontworpen voor het verwijderen van impulsruis.

Constructie Stellen we door $m(i,j)$ de grijswaarde van de mediaan in een 3×3 venster rond de te filteren pixel $A(i,j)$ voor, dan wordt de output van deze filter gegeven door:

$$A'(i,j) = m(i,j) + \mu[A(i,j)](A(i,j) - m(i,j))$$

Hierbij stelt $\mu[A(i,j)]$ de lidmaatschapsfunctie van de vaagverzameling `niet -vervuild` voor. Om $\mu[A(i,j)]$ te bepalen, wordt gekeken naar het verschil tussen de invoerpixel $A(i,j)$ en de mediaan van het 3×3 invoervenster. Deze variabele wordt $u(i,j)$ genoemd.

$$u(i,j) = |A(i,j) - m(i,j)|.$$

Dus als $u(i,j)$ een hoge waarde heeft, is de probabiliteit dat $A(i,j)$ een ruispixel is hoog. Het probleem met deze benadering is dat fijne structuren ook een hoge waarde voor $u(i,j)$ opleveren en deze dus eveneens verwijderd worden. Daarom is een bijkomende regel noodzakelijk. Deze regel maakt gebruik van de twee pixels in het 3×3 invoervenster met het kleinste verschil in absolute waarde tussen deze pixels en de invoer $A(i,j)$, dus de twee pixels waarvoor

$$|A(i,j) - A(i+k, j+l)| \qquad k,l \in [-1,1]$$

minimaal is. Noem dit verschil respectievelijk $a(i,j)$ en $b(i,j)$. We definiëren nu een nieuwe variabele $v(i,j)$ als het gemiddelde van $a(i,j)$ en $b(i,j)$, dus

$$v(i,j) = \frac{a(i,j) + b(i,j)}{2}.$$

We kunnen nu weer zeggen dat als $v(i,j)$ een grote waarde geeft, de pixel $A(i,j)$ waarschijnlijk een ruispixel is. Samengevat krijgen we volgende vaagregels:

IF $u(i,j)$ is `small` AND $v(i,j)$ is `small`, then $\mu[A(i,j)]$ is `large`
IF $u(i,j)$ is `small` AND $v(i,j)$ is `large`, then $\mu[A(i,j)]$ is `small`
IF $u(i,j)$ is `large` AND $v(i,j)$ is `small`, then $\mu[A(i,j)]$ is `small`
IF $u(i,j)$ is `large` AND $v(i,j)$ is `large`, then $\mu[A(i,j)]$ is `very small`.

Op deze manier bekomen we een lidmaatschapsfunctie $\mu[A(i,j)]$ afhankelijk van 2 veranderlijken $u(i,j)$ en $v(i,j)$. Deze functie kan benaderd worden door een trapfunctie die geoptimaliseerd wordt m.b.v. een gradiëntmethode. Als men de waarde van μ in het $(p,q)^{\text{de}}$ gebiedje noteert als $\mu_{p,q}$, dan wordt deze met een trainingsbeeld A als volgt bepaald:

$$\mu_{p,q} = \frac{\sum_{(i,j)\in G_{p,q}} [A(i,j) - m(i,j)] \cdot [D(i,j) - m(i,j)]}{\sum_{(i,j)\in G_{p,q}} [A(i,j) - m(i,j)]^2}.$$

Hierbij is D de gewenste output horende bij het trainingsbeeld A. Er wordt gesommeerd over $G_{p,q}$, de verzameling van pixels (i,j) waarvoor $u(i,j)$ tot het p^{de} en $v(i,j)$ tot het q^{de} gebiedje behoort. De grootte van een gebied werd in de implementatie 10×10 genomen.

2.2.2 Vaaglogische beslissingsgebaseerde filter - FDDF

Idee In een klassieke filter wordt iedere pixel verwerkt, ongeacht het een ruispixel betreft. Met behulp van de vaaglogica wordt nu een gewicht bepaald om de gewogen som te nemen van de klassieke uitvoer en de te filteren pixel. Dit is de idee achter de fuzzy decision directed filter[2]. Deze filter is geschikt voor het verwijderen van impulsruis.

Constructie De uitvoer van deze filter is opnieuw een gewogen som van de invoer en de uitvoer van een klassieke filter:

$$A'(i,j) = \mu[A(i,j)] \times k(i,j) + (1 - \mu[A(i,j)]) \times A(i,j).$$

Hierbij is $\mu[A(i,j)]$ de mate waarin de pixel $A(i,j)$ ruis is en $k(i,j)$ de uitvoer van een klassieke filter. Om nu $\mu[A(i,j)]$ te bepalen wordt gebruik gemaakt van een 3×3 invoervenster. Benoem de grijswaarden van de pixels van het venster zoals in figuur 2.5 en noem $d_k = A(i,j) - P_k$. Dan is de kans dat $A(i,j)$ vervuild is met ruis groot, als d_i groot is. Er worden 4 richtingen onderzocht: de horizontale, de verticale, de diagonale van rechtsboven naar linksonder en de diagonale van rechtsonder naar links boven. De regels worden aldus:

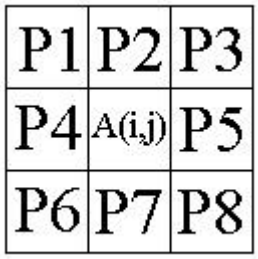
IF d_{k1} AND d_{k2} is `high positive` THEN $\mu_k[A(i,j)]$ is `high`
IF d_{k1} AND d_{k2} is `high negative` THEN $\mu_k[A(i,j)]$ is `high`
ELSE $\mu_k[A(i,j)]$ is `low`.

Hierbij is $k = horizontaal, verticaal, diagonaal_1, diagonaal_2$. Aangezien op deze manier ook fijne lijntjes aanzien worden als ruis, worden nu de richtingen horizontaal en verticaal weer gecombineerd tot de regels:

IF $\mu_k[A(i,j)]$ AND $\mu_l[A(i,j)]$ is `high positive` THEN $\mu_{k,l}[A(i,j)]$ is `high`
IF $\mu_k[A(i,j)]$ AND $\mu_l[A(i,j)]$ is `high negative` THEN $\mu_{k,l}[A(i,j)]$ is `high`
ELSE $\mu_{k,l}[A(i,j)]$ is `low`

met $k = horizontaal$ en $l = verticaal$.
Hetzelfde wordt nu gedaan voor beide diagonale richtingen. Tot slot worden de 2 bekomen $\mu_{k,l}[A(i,j)]$ op dezelfde manier gecombineerd tot de definitieve $\mu[A(i,j)]$. Wanneer voor de conjunctie de minimum-operator gebruikt wordt en in de veronderstelling dat de lidmaatschapsfunctie f van `high` strikt stijgend is, dan kan $\mu[A(i,j)]$ als volgt bepaald worden:

$$\mu[A(i,j)] = f[min(d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8)].$$



Figuur 2.5: Nummering van de pixels in het 3×3 invoervenster

Op deze manier wordt een pixel als ruis aanzien, wanneer het verschil met ALLE omliggende pixels groot is. Wanneer dus 2 pixels in hetzelfde invoervenster geïnfecteerd zijn met zout&peper-ruis, dan wordt geen van de 2 pixels als ruis aanzien. Dit heeft nefaste gevolgen voor de performantie van de filter.
Als lidmaatmaatschapsfunctie voor `high`, werd in de implementatie volgende functie gebruikt:

$$\mu_{\text{high}}(x) = \begin{cases} 0 & x < a \\ \frac{x-a}{b-a} & a \leq x \leq b \\ 1 & b \leq x \end{cases}$$

De parameters a en b zijn vrij instelbaar, doch de resultaten veranderen weinig afhankelijk van deze parameters. Een beeld gefilterd met de FDDF-filter wordt getoond in figuur 2.6.

Figuur 2.6: FDDF-filter toegepast op Lena vervuild met zout&peper-ruis: (a)Origineel beeld; (b)$\delta = 0.05$; (c)$\delta = 0.25$; (d)$\delta = 0.50$.

2.2.3 Gewogen vaag gemiddelde filter - WFM

Idee De weighted fuzzy mean filter[3] werd ontwikkeld voor het filteren van beelden die sterk vervuild zijn met zout&peper ruis. Het algoritme zou gemakkelijk in hardware te implementeren zijn. Het maakt gebruik van een kennisbank die ofwel statisch door experten of met behulp van voorbeeldbeelden, ofwel dynamisch met behulp van het histogram van het beeld opgebouwd wordt. Voor de opbouw van de dynamische databank wordt gebruik gemaakt van het originele beeld, dus deze filter zou goede prestaties moeten leveren.

Constructie van de kennisbank De kennisbank bestaat uit 9 getallen. Deze 9 getallen worden gebruikt als parameters α, β en m voor de lidmaatschapsfuncties van de vaagverzamelingen `dark`, `medium` en `bright`, die volgende vorm hebben:

$$\mu(x) = \begin{cases} L(\frac{m-x}{\alpha}) & x \leq m \\ R(\frac{x-m}{\beta}) & x \geq m, \end{cases}$$

met $L(y) = R(y) = \max(0, 1-y)$. Eventueel kunnnen meer vaagverzamelingen toegevoegd worden. Het algoritme om deze parameters dynamisch te bepalen maakt gebruik van het histogram van de beeld. Het histogram van een beeld met grijswaarden in het interval $[0, 255]$ is een functie

$$p(s_k) = \frac{n_k}{n},$$

waarbij s_k de k$^{\text{de}}$ grijswaarde van het beeld is, n_k het aantal pixels in het beeld met grijswaarde s_k is en n het totale aantal pixels in het beeld is.

Stap 1 Bepaal de intervallen $[DK_{begin}, DK_{end}]$, $[MD_{begin}, MD_{end}]$ en $[BR_{begin}, BR_{end}]$ voor respectievelijk de vaagverzamelingen `dark`, `medium` en `bright`.
Stap 1.1
$DK_{end} = \lfloor \frac{L-1}{N_f} \rfloor$, $BR_{begin} = (N_f - 1)\lfloor \frac{L-1}{N_f} \rfloor$, $MD_{begin} = DK_{end} - left\_overlap$ en $MD_{end} = BR_{begin} + right\_overlap$, waarbij N_f het aantal vaagverzamelingen is en $left\_overlap$ en $right\_overlap$ het overlappingsgebied tussen de vaagverzamelingen aanduiden.
Stap 1.2
Stel DK_{begin} de eerste s_k zodat $n_k > t$ van 0 tot DK_{end} met t een drempelwaarde.
Stap 1.3
Stel BR_{end} de laatste s_k zodat $n_k > t$ van BR_{begin} tot 255.

Stap 2 Zoek het punt s_k met de maximumwaarde voor $p(s_k)$ in het interval $[DK_{begin}, DK_{einde}]$, genereer dan de lidmaatschapsfunctie f_{DK} van de vaagverzameling `dark` met behulp van de volgende stappen:
Stap 2.1:$m_{DK} \leftarrow s_k$
Stap 2.2:$\alpha_{DK} \leftarrow m_{DK} - DK_{begin}$
Stap 2.3:$\beta_{DK} \leftarrow DK_{einde} - m_{DK}$.

Stap 3 Herhaal stap 2 voor de vaagverzamelingen `medium` en `bright`. Op deze manier wordt de kennisbank gecreëerd. Aangezien de kennisbank enorm klein is in vergelijking met het beeld, is de kans dat een fout gemaakt wordt tijdens het doorzenden ervan relatief klein en met geschikte foutcorrigerende codes kan deze kans tot 0 herleid worden.

Constructie van de eigenlijke filter Het eigenlijke filteren gebeurt in 2 stappen: drie vaag gemiddelde processen en een beslissingsproces. Elk vaag gemiddelde proces bepaalt een mogelijke waarde voor $A'(i,j)$. Hierbij wordt gebruik gemaakt van de gewogen gemiddelde benadering in een 3×3 venster met $A(i,j)$ als centrale pixel. De gewichten worden bepaald aan de hand van de lidmaatschapsfuncties die door de kennisbank bepaald worden. Het beslissingsproces bepaalt dan aan de hand van een vage schatter welke waarde de uiteindelijke waarde voor $A'(i,j)$ wordt.

Vaag gemiddelde proces Er is een vaag gemiddelde proces voor elk van de drie vaagverzamelingen. Als voorbeeld wordt hier het algoritme voor de vaagverzameling `dark` gegeven:

IF

$$\sum_{k=-1}^{1} \sum_{l=-1}^{1} f_{DK}(A(i+k, j+l)) \neq 0$$

THEN

$$\bar{y}_{DK}(i,j) \leftarrow \frac{\sum_{k=-1}^{1} \sum_{l=-1}^{1} f_{DK}(A(i+k, j+l)) \times A(i+k, j+l)}{\sum_{k=-1}^{1} \sum_{l=-1}^{1} f_{DK}(A(i+k, j+l))}$$

ELSE

$$\bar{y}_{DK}(i,j) \leftarrow 0.$$

Na het berekenen van $\bar{y}_{DK}(i,j)$, $\bar{y}_{MD}(i,j)$ en $\bar{y}_{BR}(i,j)$ wordt nu het beslissingsproces opgeroepen om te bepalen wat de definitieve output moet worden.

Beslissingsproces Het beslissingsproces maakt gebruik van een vage schatter $f_{LR\_E}(.)$ die als volgt gedefinieerd wordt:

$$f_{LR\_E}(A(i,j)) = \frac{\sum_{k=-1}^{1}\sum_{l=-1}^{1} f_{LR\_I}(A(i+k,j+l)) \times A(i+k,j+l)}{\sum_{k=-1}^{1}\sum_{l=-1}^{1} f_{LR\_I}(A(i+k,j+l))}.$$

Bij deze implementatie van de filter wordt f_{LR_I} als volgt gedefinieerd:

$$f_{LR\_I}(x) = \begin{cases} 0 & x < DK_{begin} \\ 1 & DK_{begin} \leq x \leq BR_{end} \\ 0 & x > BR_{end}. \end{cases}$$

Nu kan het eigenlijke beslissingsproces opgeroepen worden:

$$\text{IF}$$

$$|\bar{y}_{DK}(i,j) - f_{LR\_E}(A(i,j)| < |\bar{y}_{MD}(i,j) - f_{LR\_E}(A(i,j)|$$

$$\text{THEN}$$

$$A'(i,j) \leftarrow \bar{y}_{DK}(i,j)$$

$$\text{ELSE}$$

$$A'(i,j) \leftarrow \bar{y}_{MD}(i,j)$$

$$\text{IF}$$

$$|\bar{y}_{BR}(i,j) - f_{LR\_E}(A(i,j)| < |A(i,j) - f_{LR\_E}(A(i,j)|$$

$$\text{THEN}$$

$$A'(i,j) \leftarrow \bar{y}_{BR}(i,j).$$

Een beeld gefilterd met de WFM-filter wordt getoond in figuur 2.7.

Figuur 2.7: WFM-filter met t=10 toegepast op Lena vervuild met zout&peper-ruis: (a)Origineel beeld; (b)$\delta = 0.05$; (c)$\delta = 0.25$; (d)$\delta = 0.50$.

2.2.4 Vaaglogische meerstaps mediaanfilter - FMMF

Idee De FMMF[4] filter is gebaseerd op de multilevel median filter en maakt gebruik van vage regels om de performantie te verhogen. De multilevel median filter werd ontworpen om impulsruis te verwijderen en werkt met een $l \times l$ venster met de te filteren pixel als centrale pixel. In tegenstelling tot de gewone mediaanfilter, waar de mediaan van het invoervenster wordt berekend, wordt hier voor elk van de 4 richtingen horizontaal, verticaal en de 2 diagonalen de mediaan berekend om zo in verschillende fasen de uiteindelijke waarde te bepalen.

Constructie Definieer vier deelvensters van het oorspronkelijke $(2l+1) \times (2l+1)$ invoervenster $W_{(i,j)}$ als volgt:

$$\begin{aligned} W_{(i,j),1} &= \{A(i, j+k) : -l \leq k \leq l\} \\ W_{(i,j),2} &= \{A(i+k, j+k) : -l \leq k \leq l\} \\ W_{(i,j),3} &= \{A(i+k, j) : -l \leq k \leq l\} \\ W_{(i,j),4} &= \{A(i+k, j-k) : -l \leq k \leq l\} \end{aligned}$$

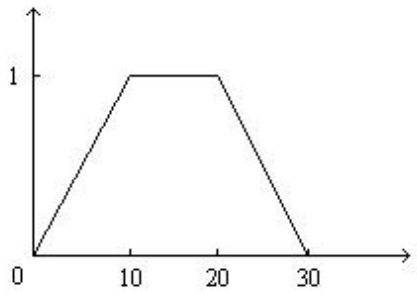
Stel nu $\text{med}_x(i,j)$, voor $x = 1, 2, 3, 4$ de mediaan van ieder deelvenster. Dan kunnen 2 variabelen gedefinieerd worden:

$$\begin{aligned} \text{med}_{max}(i,j) &= \max_{x=1}^{4} \text{med}_x(i,j) \\ \text{med}_{min}(i,j) &= \min_{x=1}^{4} \text{med}_x(i,j) \end{aligned}$$

Dan wordt de output van de multilevel mediaan filter:

$$A'(i,j) = \text{med}(\text{med}_{max}(i,j), \text{med}_{min}(i,j), A(i,j)).$$

Om de performantie te verhogen worden nu nog 2 termen toegevoegd, gebruik makend van technieken uit de vaaglogica. Als elementen van de vaagverzameling krijgen we de verschillen tussen de mediaan van ieder deelvenster en elk van de pixels $w_{xk}, x = 1, \ldots, 4; -l \leq k \leq l$ in ieder deelvenster. Gebruiken we d_{xy} om het verschil tussen de mediaan van deelvenster x en pixel w_{xk} uit te drukken, dan kunnen we deze informatie gebruiken om de functie C te bepalen die aangeeft of de mediaan van ieder deelvenster een goede keuze zou zijn, aan de hand van volgende regels:
IF d_{xy} is `high` THEN $C(d_{xy})$ is `low`
IF $d_{xy} \approx 0$ THEN $C(d_{xy})$ is `low`
IF d_{xy} is `medium` THEN $C(d_{xy})$ is `high`



Figuur 2.8: Lidmaatschapsfunctie voor de functie C, voorgesteld door de auteur.

De vorm van de lidmaatschapsfunctie die voorgesteld werd door de auteur, wordt getoond in figuur 2.8. Met de functie C kan een nieuwe functie $s_x(i,j)$ voor ieder deelvenster gedefinieerd worden:

$$s_x(i,j) = \sum_{j=l-1}^{l+1} C_{\text{med}}(d_{xy}) \text{ voor } x = 1, \ldots, 4.$$

De uitvoer van deze filter wordt dan gegeven door:

$$A'(i,j) = \text{med}(\text{med}_{max}(i,j), \text{med}_{min}(i,j), A(i,j), s_{\text{max1}}(i,j), s_{\text{max2}}(i,j)).$$

Hierbij zijn $s_{\text{max1}}(i,j)$ en $s_{\text{max2}}(i,j)$ de grootste 2 waarden van $s_x(i,j)$. Een beeld gefilterd met de FMMF-filter wordt getoond in figuur 2.9.

Figuur 2.9: FMMF-filter met venstergrootte 7×7 toegepast op Lena vervuild met zout&peper-ruis: (a)Origineel beeld; (b)$\delta = 0.05$; (c)$\delta = 0.25$; (d)$\delta = 0.50$.

2.2.5 Enkele filters gebaseerd op gewogen som

Deze filters uit [5] hebben allemaal dezelfde structuur, ze hebben namelijk alle als output een genormaliseerde gewogen som van alle pixels uit het invoervenster van grootte $r \times r$ met de te filteren pixel als centrale pixel:

$$A'(i,j) = \frac{\sum_{-l \leq r,s \leq l} F[A(i+r, j+s)] \times A(i+r, j+s)}{\sum_{-l \leq r,s \leq l} F[A(i+r, j+s)]}.$$

$F[A(i+r, j+s)]$ wordt de vensterfunctie genoemd en is een functie die de filter volledig bepaalt. Als invoervenster kan eigenlijk om het even welk rechthoekig venster gebruikt worden, maar de voorkeur wordt toch gegeven aan een vierkant venster.
Een deel van de filters die hierna besproken worden is gebaseerd op de klassieke mediaanfilter, die ontworpen werd voor het verwijderen van impulsruis. Een ander deel is gebaseerd op de klassieke moving average filter, die een betere oplossing biedt voor het verwijderen van gaussische ruis. De mediaanfilter en de moving average filter werden in een vorig deel van deze scriptie reeds besproken.

Gaussiaanse vaaglogische filter met mediaancentrum - GMED

Idee Bij deze filter wordt gebruik gemaakt van de gaussische distributiefunctie. In tegenstelling tot de gewone gaussische filter, waar de te filteren pixel als centrum gebruikt wordt, wordt hier de mediaan van het invoervenster als centrum gebruikt. Hoe groter het verschil tussen een pixel uit het invoervenster en de mediaan ervan, hoe kleiner het gewicht dat toegekend wordt aan deze pixel. Gaussische filters hebben vooral als gevolg dat het beeld waziger wordt.

Constructie In $\mathbb{R}^2$ wordt de Gauss-functie gegeven door

$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

met σ^2 de variantie. De vensterfunctie van de GMED-filter is dan:

$$F_{gmed}[A(i+r, j+s)] = e^{-\frac{1}{2}[\frac{A(i+r,j+s)-A_{\text{med}}(i,j)}{\sigma(i,j)}]^2}$$

Hierbij is $A_{\text{med}}(i,j)$ de mediaan van het invoervenster en $\sigma(i,j)^2$ de variantie binnen het invoervenster. Voor de volledigheid herhalen we nog eens de manier waarop de variantie bepaald wordt.

$$\sigma(i,j)^2 = \frac{\sum_{r=-l}^{l} \sum_{s=-l}^{l} (A(i+r, j+s) - \mu(i,j))^2}{(2l+1) \times (2l+1)}$$

met $\mu(i,j)$ het gemiddelde van de waarden in het invoervenster met centrale pixel $A(i,j)$. Een beeld gefilterd met de GMED-filter wordt getoond in figuur 2.10.

Symmetrische driehoeksvormige vaaglogische filter met mediaancentrum - TMED

Idee De TMED-filter kent een groter gewicht toe aan pixels waarvan de waarde dichter bij de mediaan van het invoervenster gelegen is.

Constructie De vensterfunctie wordt hier

$$F_{tmed}[A(i+r,j+s)] = \begin{cases} 1 - \frac{|A(i+r,j+s)-A_{\text{med}}(i,j)|}{A_{\text{mm}}(i,j)} & |A(i+r,j+s)- \\ & \quad A_{\text{med}}(i,j)| \leq A_{\text{mm}}(i,j) \\ 1 & A_{\text{mm}}(i,j) = 0 \end{cases}$$

met $A_{\text{mm}}(i,j) = \max[A_{\max}(i,j) - A_{\text{med}}(i,j), A_{\text{med}}(i,j) - A_{\min}(i,j)]$
$A_{\max}(i,j)$,$A_{\min}(i,j)$ en $A_{\text{med}}(i,j)$ zijn respectievelijk het maximum, het minimum en de mediaan van het invoervenster met centrale pixel $A(i,j)$. Een beeld gefilterd met de TMED-filter wordt getoond in figuur 2.11.

Asymmetrische driehoeksvormige vaaglogische filter met mediaancentrum - ATMED

Idee Het idee is in feite hetzelfde als bij de TMED-filter, maar in tegenstelling tot de TMED-filter is de vensterfuncite van de ATMED-filter asymmetrisch. De graad van asymmetrie is afhankelijk van het verschil tussen het verschil tussen het maximum en de mediaan en het verschil tussen de mediaan en het minimum.

Constructie De vensterfunctie wordt hier:

$$F_{atmed}[A(i+r,j+s)] = \begin{cases} 1 - \frac{A_{\text{med}}(i,j)-A(i+r,j+s)}{A_{\text{med}}(i,j)-A_{\min}(i,j)} & A_{\min} \leq A(i+r,j+s) \\ & \quad \leq A_{\text{med}}(i,j) \\ 1 - \frac{A(i+r,j+s)-A_{\text{med}}(i,j)}{A_{\max}(i,j)-A_{\text{med}}(i,j)} & A_{\text{med}} \leq A(i+r,j+s) \\ & \quad \leq A_{\max}(i,j) \\ 1 & A_{\text{med}}(i,j) = A_{\min}(i,j) \\ & \text{ of } A_{\text{med}}(i,j) = A_{\max}(i,j) \end{cases}$$

Een beeld gefilterd met de ATMED-filter wordt getoond in figuur 2.12.

Figuur 2.10: GMED-filter met venstergrootte 3×3 toegepast op Lena vervuild met zout&peper-ruis: (a)Origineel beeld; (b)$\delta = 0.05$; (c)$\delta = 0.25$; (d)$\delta = 0.50$.

Figuur 2.11: TMED-filter met venstergrootte 3×3 toegepast op Lena vervuild met zout&peper-ruis: (a)Origineel beeld; (b)$\delta = 0.05$; (c)$\delta = 0.25$; (d)$\delta = 0.50$.

Figuur 2.12: ATMED-filter met venstergrootte 3×3 toegepast op Lena vervuild met zout&peper-ruis: (a)Origineel beeld; (b)$\delta = 0.05$; (c)$\delta = 0.25$; (d)$\delta = 0.50$.

Gaussiaanse vaaglogische filter met gemiddelde waarde centrum - GMAV

Idee Analoog aan de GMED-filter, maar nu wordt niet het verschil met de mediaan, maar het verschil met het gemiddelde van het invoervenster in rekening gebracht.

Constructie

$$F_{gmav}[A(i+r,j+s)] = e^{-\frac{1}{2}[\frac{A(i+r,j+s)-A_{\text{mav}}(i,j)}{\sigma(i,j)}]^2}$$

Hierbij zijn A_{mav} en σ^2 respectievelijk het gemiddelde van het invoervenster en de variantie binnen het invoervenster. Een beeld gefilterd met de GMAV-filter wordt getoond in figuur 2.13.

Symmetrische driehoeksvormige vaaglogische filter met gemiddelde waarde centrum - TMAV

Idee De TMAV-filter kent een groter gewicht toe aan pixels die dichter bij de gemiddelde waarde van alle grijswaarden uit het invoervenster gelegen zijn.

Constructie De vensterfunctie wordt

$$F_{tmav}[A(i+r,j+s)] = \begin{cases} 1 - \frac{|A(i+r,j+s)-A_{\text{mav}}(i,j)|}{A_{\text{mv}}(i,j)} & |A(i+r,j+s) - A_{\text{mav}}(i,j)| \\ & \leq A_{\text{mv}}(i,j) \\ 1 & A_{\text{mv}} = 0 \end{cases}$$

Hierbij is $A_{\text{mv}} = \max[A_{\text{max}}(i,j) - A_{\text{mav}}(i,j), A_{\text{mav}}(i,j) - A_{\text{min}}(i,j)]]$
A_{max}, A_{min} en A_{mav} zijn respectievelijk het minimum, het maximum en het gemiddelde van de grijswaarden van de pixels uit het invoervenster. Een beeld gefilterd met de TMAV-filter wordt getoond in figuur 2.14.

Asymmetrische driehoeksvormige vaaglogische filter met gemiddelde waarde centrum - ATMAV

Idee Er wordt een groter gewicht toegekend aan pixels die dichter bij de gemiddelde waarde van het invoervenster liggen, maar deze waarde hangt ook af van de ligging van de pixel, namelijk het gegeven of de pixel groter of kleiner is dan het gemiddelde.

Figuur 2.13: GMAV met venstergrootte 3×3 toegepast op Lena vervuild met gaussische ruis: (a)Origineel beeld; (b)$\sigma = 7$; (c)$\sigma = 15$; (d)$\sigma = 25$.

Figuur 2.14: TMAV-filter met venstergrootte 3×3 toegepast op Lena vervuild met gaussische ruis: (a)Origineel beeld; (b)$\sigma = 7$; (c)$\sigma = 15$; (d)$\sigma = 25$.

Constructie De vensterfunctie wordt hier:

$$F_{atmav}[A(i+r,j+s)] = \begin{cases} 1 - \frac{A_{\text{mav}}(i,j) - A(i+r,j+s)}{A_{\text{mav}}(i,j) - A_{\text{min}}(i,j)} & A_{\text{min}} \leq A(i+r,j+s) \\ & \leq A_{\text{mav}}(i,j) \\ 1 - \frac{A(i+r,j+s) - A_{\text{mav}}(i,j)}{A_{\text{max}}(i,j) - A_{\text{mav}}(i,j)} & A_{\text{mav}} \leq A(i+r,j+s) \\ & \leq A_{\text{max}}(i,j) \\ 1 & A_{\text{mav}}(i,j) = A_{\text{min}}(i,j) \\ & \text{of } A_{\text{mav}}(i,j) = A_{\text{max}}(i,j) \end{cases}$$

Een beeld gefilterd met de ATMAV-filter wordt getoond in figuur 2.15.

Afnemend gewicht vaaglogische filter met gemiddelde waarde centrum - DWMAV

Idee Het idee achter deze filter is eigenlijk heel eenvoudig. Hoe verder de pixel gelegen is van het centrum van het invoervenster, hoe minder belang aan deze pixel gehecht wordt.

Constructie De vensterfunctie wordt als volgt gedefinieerd

$$F_{dwmav}[A(i+r,j+s)] = 1 - \frac{\max(|r|,|s|)}{l+t}$$

met de venstergrootte $(2l+1) \times (2l+1)$
t is een drempelwaarde die de richtingscoëfficiënt van de driehoekvormige functie bepaalt. De waarden die gebruikt worden voor t zijn 1,2 en 3. Om het onderscheid te maken, zullen we naar deze filters verwijzen met respectievelijk DWMAV1, DWMAV2 en DWMAV3. Een beeld gefilterd met de DWMAV1-filter wordt getoond in figuur 2.16. Een beeld gefilterd met de DWMAV2-filter wordt getoond in figuur 2.17. Een beeld gefilterd met de DWMAV3-filter wordt getoond in figuur 2.18.

Figuur 2.15: ATMAV-filter met venstergrootte 3×3 toegepast op Lena vervuild met gaussische ruis: (a)Origineel beeld; (b)$\sigma = 7$; (c)$\sigma = 15$; (d)$\sigma = 25$.

Figuur 2.16: DWMAV1-filter met venstergrootte 3×3 toegepast op Lena vervuild met gaussische ruis: (a)Origineel beeld; (b)$\sigma = 7$; (c)$\sigma = 15$; (d)$\sigma = 25$.

Figuur 2.17: DWMAV2-filter met venstergrootte 3×3 toegepast op Lena vervuild met gaussische ruis: (a)Origineel beeld; (b)$\sigma = 7$; (c)$\sigma = 15$; (d)$\sigma = 25$.

Figuur 2.18: DWMAV3-filter met venstergrootte 3×3 toegepast op Lena vervuild met gaussische ruis: (a)Origineel beeld; (b)$\sigma = 7$; (c)$\sigma = 15$; (d)$\sigma = 25$.

2.3 Vaagfilters

2.3.1 Russo-ramponi filter - RR

Idee De RR-filter[6] werd ontworpen voor het verwijderen van zout&peper-ruis en bestaat uit 2 stappen. Eerst wordt een mogelijke correctieterm voorgesteld met behulp van vaaglogica. Om deze correctieterm te bepalen worden verscheidene patronen binnen het 3×3 invoervenster onderzocht. Het is vooral belangrijk dat de grijswaarden van de pixels binnen een patroon niet te ver van elkaar gelegen zijn. Zo kunnen bepaalde vormen van fijne details gedetecteerd worden. In een tweede stap wordt de uiteindelijke correctieterm berekend. Hiervoor wordt gebruik gemaakt van de vaagverzameling `small`, aangezien kleine wijzigingen toch niet veel bijdragen tot het wegwerken van zout&peper ruis en bovendien randen wazig maken.

Constructie

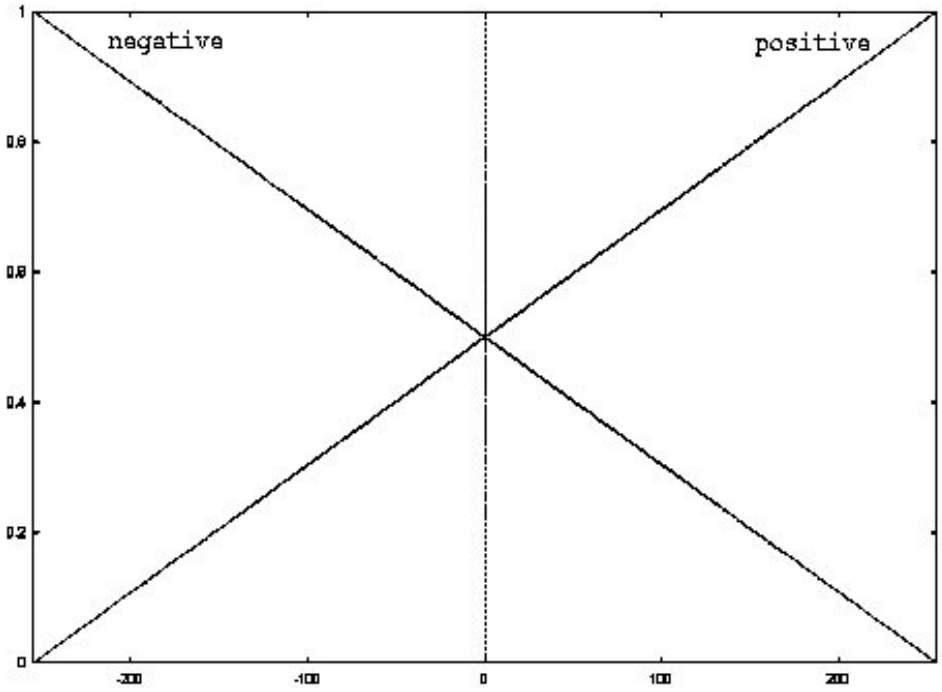
Bepalen van de mogelijke correctieterm Voor het bepalen van de correctieterm wordt gebruik gemaakt van 2 vaagverzamelingen:`positive` en `negative`. De lidmaatschapsfuncties worden getoond in figuur 2.19. De functie wordt gegeven door

$$\mu(x) = \begin{cases} 0 & x \leq c - w \\ \frac{(w-|x-c|)}{w} & c - w < x < c + w \\ 0 & c + w \leq x \end{cases}$$

Hierbij is voor `positive` $c = 255$ en $w = 510$ en voor `negative` $c = -255$ en $w = 510$. De definitieverzameling van μ zijn de verschillen in grijswaarden tussen de pixels uit het invoervenster en de centrale pixel: $d_k = P_k - A(i, j)$. Hierbij worden de pixels genummerd zoals in figuur 2.5, bij de FDDF-filter[1]. Er worden 13 patronen onderzocht:

$$\begin{array}{lll} I_1 = \{d_2, d_5, d_7\} & I_2 = \{d_5, d_7, d_4\} & I_3 = \{d_7, d_4, d_2\} \\ I_4 = \{d_4, d_2, d_5\} & I_5 = \{d_1, d_3, d_8, d_6\} & I_6 = \{d_1, d_2, d_3, d_5\} \\ I_7 = \{d_2, d_3, d_5, d_8\} & I_8 = \{d_3, d_5, d_8, d_7\} & I_9 = \{d_5, d_8, d_7, d_6\} \\ I_{10} = \{d_8, d_7, d_6, d_4\} & I_{11} = \{d_7, d_6, d_4, d_1\} & I_{12} = \{d_6, d_4, d_1, d_2\} \\ I_{13} = \{d_4, d_1, d_2, d_3\} & & \end{array}$$

[1]Let wel op: bij de FDDF-filter is $d_k = A(i, j) - P_k$



Figuur 2.19: Lidmaatschapsfuncties van positive en negative

En de vaagregels worden dan:
IF $\forall d_k \in I_l : d_k$ is `positive` THEN $y(i,j)$ is `positive`
IF $\forall d_k \in I_l : d_k$ is `negative` THEN $y(i,j)$ is `negative`
voor $l = 1 \ldots 13$
Op basis van deze regels wordt een mogelijke correctieterm $y(i,j)$ bepaald:

$$\lambda_1 = \max\{\min\{\mu_{PO}(d_k); d_k \in I_l\}; l = 1, \ldots, 13\}$$

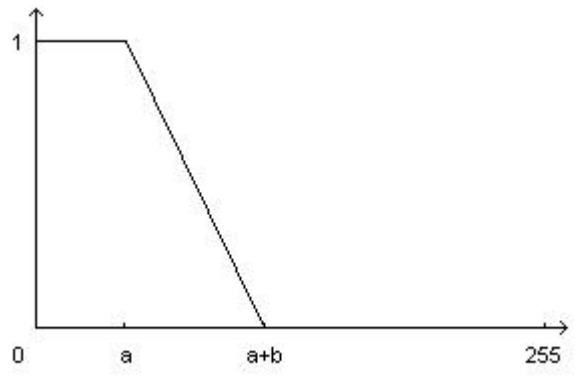
$$\lambda_2 = \max\{\min\{\mu_{NE}(d_k); d_k \in I_l\}; l = 1, \ldots, 13\}$$

$$\lambda_0 = \max\{0, 1 - \lambda_1 - \lambda_2\}$$

$$y(i,j) = 255\frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2 + \lambda_0}$$

Hierbij zijn μ_{PO} en μ_{NE} respectievelijk de lidmaatschapsfuncties van de vaagverzamelingen `positive` en `negative`.

Bepalen van de uiteindelijke correctieterm Om de uiteindelijke correctieterm te bepalen worden kleine wijzigingen weggewerkt. Dit wordt



Figuur 2.20: Lidmaatschapsfunctie van small

bereikt door een vaagverzameling `small` in te voeren, waarvan de lidmaatschapsfunctie getoond wordt in figuur 2.20. Het functievoorschrift is:

$$\mu_{\text{small}}(x) = \begin{cases} 1 & x \leq a \\ \frac{a+b-x}{b} & a < x \leq a+b \\ 0 & a+b < x \end{cases}$$

De keuze van de parameters a en b is niet kritiek, maar typische waarden zijn $a = 40$ en $b = 32$. De uiteindelijke waarde voor de gefilterde pixel $A'(i,j)$ wordt nu:

$$A'(i,j) = A(i,j) + y(i,j)(1 - \mu_{small}(|y(i,j)|))$$

Een beeld gefilterd met de RR-filter wordt getoond in figuur 2.21.

Figuur 2.21: RR-filter met $a = 42$ en $b = 30$ toegepast op Lena vervuild met zout&peper-ruis: (a)Origineel beeld; (b)$\delta = 0.05$; (c)$\delta = 0.25$; (d)$\delta = 0.50$

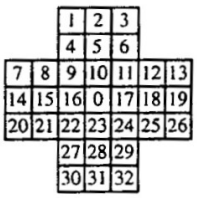
2.3.2 Tweestaps firefilter - DS-FIRE

Idee De DS-FIRE-filter[7] is lid van de familie van FIRE[2]-filters. FIRE-filters bestaan uit 2 stappen: in de eerste stap wordt een mogelijke correctie-term berekend en in de tweede stap wordt deze term gecorrigeerd, door kleine wijzigingen ongedaan te maken. FIRE-filters zijn speciaal ontworpen voor het verwijderen van zout&peper-ruis. Bij de constructie zullen we zien dat deze filter volledig analoog opgebouwd wordt als de RR-filter. In feite is de RR-filter ook een FIRE-filter, ondanks het feit dat er geen ELSE-statement voorkomt in de vaagregels. Het enige feitelijke verschil tussen de RR-filter en de DSFIRE-filter is dat er nu een 7×7 kruisvormig venster gebruikt wordt.

Constructie

Bepalen van de voorlopige uitvoer De pixels worden genummerd zoals in figuur 2.22. Noemen we nu de verschillen $d_k = x_k - x_0$. Hierbij is x_k de pixel met label k volgens de nummering in figuur 2.22 en is x_0 de te filteren pixel $A(i, j)$. Om de regels op te stellen wordt gebruik gemaakt van 3 vaagverzamelingen:`positive`, `negative` en `zero`, waarvan de lidmaatschaps-functies afgebeeld worden in figuur 2.23. De regelbank bestaat dan uit een reeks IF-THEN-ELSE regels van de vorm:
IF $\forall d_k \in I_l : d_k$ is `positive` THEN $y(i, j)$ is `positive`
IF $\forall d_k \in I_l : d_k$ is `negative` THEN $y(i, j)$ is `negative`
ELSE $y(i, j)$ is `zero`

[2]Fuzzy Inference Ruled by Else-action



Figuur 2.22: Nummering van de pixels voor de DS-FIRE-filter

Hierbij zijn

$$I_1 = \{d_{10}, d_{16}, d_{23}\} \qquad I_2 = \{d_{10}, d_{16}, d_{17}\}$$
$$I_3 = \{d_{10}, d_{17}, d_{23}\} \qquad I_4 = \{d_{16}, d_{17}, d_{12}\}$$
$$I_5 = \{d_9, d_{11}, d_{24}, d_{22}\} \qquad I_6 = \{d_{10}, d_{11}, d_{17}, d_{24}\}$$
$$I_7 = \{d_{10}, d_9, d_{16}, d_{22}\} \qquad I_8 = \{d_9, d_{10}, d_{17}, d_{24}\}$$
$$I_9 = \{d_4, d_5, d_6, d_{12}, d_{18}, d_{25}\} \qquad I_{10} = \{d_1, d_2, d_3, d_{13}, d_{19}, d_{26}\}$$
$$I_{11} = \{d_{16}, d_9, d_{10}, d_{11}\} \qquad I_{12} = \{d_{16}, d_{22}, d_{23}, d_{24}\}$$
$$I_{13} = \{d_{22}, d_{16}, d_{10}, d_{11}\} \qquad I_{14} = \{d_{21}, d_{15}, d_8, d_4, d_5, d_6\}$$
$$I_{15} = \{d_{20}, d_{14}, d_7, d_1, d_2, d_3\} \qquad I_{16} = \{d_9, d_{16}, d_{22}, d_{23}\}$$
$$I_{17} = \{d_{23}, d_{24}, d_{17}, d_{11}\} \qquad I_{18} = \{d_9, d_{16}, d_{23}, d_{24}\}$$
$$I_{19} = \{d_8, d_{15}, d_{21}, d_{27}, d_{28}, d_{29}\} \qquad I_{20} = \{d_7, d_{14}, d_{20}, d_{30}, d_{31}, d_{32}\}$$
$$I_{21} = \{d_{22}, d_{23}, d_{24}, d_{17}\} \qquad I_{22} = \{d_9, d_{10}, d_{11}, d_{17}\}$$
$$I_{23} = \{d_{22}, d_{23}, d_{17}, d_{11}\} \qquad I_{24} = \{d_{27}, d_{28}, d_{29}, d_{25}, d_{18}, d_{12}\}$$
$$I_{25} = \{d_{30}, d_{31}, d_{32}, d_{26}, d_{19}, d_{13}\}$$

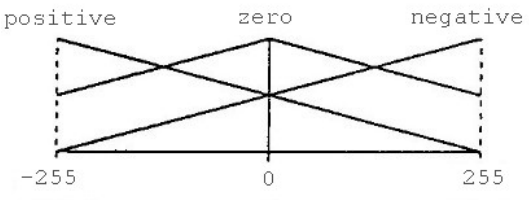
Dit komt overeen met de patronen getoond in figuur 2.24. Op deze manier bekomt men 50 IF-THEN-regels en 1 IF-THEN-ELSE-regel. Op basis van deze regels wordt dan de mogelijke correctieterm $y(i, j)$ berekend:

$$\lambda_1 = \max\{\min\{\mu_{PO}(d_k); d_k \in I_l\}; l = 1, \ldots, 25\}$$

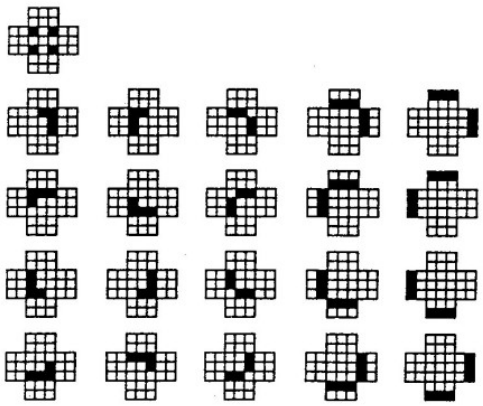
$$\lambda_2 = \max\{\min\{\mu_{NE}(d_k); d_k \in I_l\}; l = 1, \ldots, 25\}$$

$$y(i, j) = 255(\lambda_1 - \lambda_2)$$

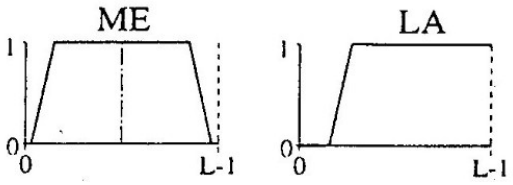
Bepalen van de uiteindelijke uitvoer Om de uiteindelijke uitvoer van de filter te bepalen, wordt de bekomen waarde zodanig bewerkt dat het behoud van fijne details maximaal is. Hiervoor wordt gebruik gemaakt van



Figuur 2.23: Lidmaatschapsfuncties van positive, negative en zero



Figuur 2.24: Patronen die worden gecontroleerd, bevat in de verzamelingen I_l



Figuur 2.25: Lidmaatschapsfunctie van large en medium

twee vaagverzamelingen: `large` en `medium`, waarvan de lidmaatschapsfuncties getoond worden in figuur 2.25. De uiteindelijke uitvoer van de filter wordt:

$$A'(i,j) = y(i,j)\max\{\mu_{LA}(|y(i,j)|), 1 - \mu_{ME}(A(i,j))\}$$

Een beeld gefilterd met de DSFIRE-filter wordt getoond in figuur 2.26.

Figuur 2.26: DSFIRE-filter toegepast op Lena vervuild met zout&peper-ruis: (a)Origineel beeld; (b)$\delta = 0.05$; (c)$\delta = 0.25$; (d)$\delta = 0.50$

2.3.3 Iteratieve vaaglogisch gebaseerde filter - IFCF

Idee De IFCF-filter[8] verwijdert impuls ruis, verzacht gaussische ruis en behoudt de fijne details in een beeld. Hiervoor wordt gebruik gemaakt van het FIRE-mechanisme. Het grijswaarden-interval wordt opgesplitst in 7 delen en ieder deel zorgt voor een gewicht in de uiteindelijke correctieterm. Dit gewicht wordt bepaald door de verschillen tussen de te filteren pixel en de omliggende pixels in rekening te brengen.

Constructie Er wordt gebruik gemaakt van 9 verschillende vaagverzamelingen: `negative bright`(NB), `negative medium`(NM), `negative small`(NS), `positive small`(PS), `positive medium`(PM), `positive bright`(PB), `zero`(Z), `more` en `more'`. De lidmaatschapsfuncties van de eerste 6 en van `zero` zijn triangulair, zoals getoond in figuur 2.27, de lidmaatschapsfuncties van `more` en `more'` zijn van het S-type, zoals getoond in figuur 2.28 en worden als volgt gedefinieerd:

$$\mu_{\text{more}}(x) = \frac{1}{1 + e^{-(\alpha_1 x - \beta_1)}}$$

$$\mu_{\text{more'}}(x) = \frac{1}{1 + e^{-(\alpha_2 x - \beta_2)}} \quad \alpha_2 < \alpha_1$$

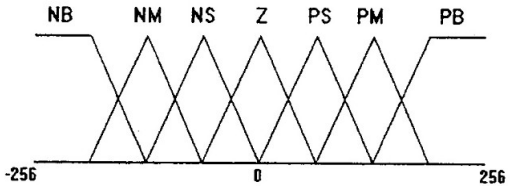
Met deze vaagverzamelingen kunnen 6 IF-THEN-regels en 1 IF-THEN-ELSE-regel gevormd worden:

IF `more` d_k are `negative bright` THEN y is `negative bright`
IF `more` d_k are `negative medium` THEN y is `negative medium`
IF `more` d_k are `negative small` THEN y is `negative small`
IF `more` d_k are `positive small` THEN y is `positive small`
IF `more` d_k are `positive medium` THEN y is `positive medium`
IF `more` d_k are `positive bright` THEN y is `positive bright`
ELSE IF `more'` d_k are `zero` THEN y is gem(d_k)

Hierbij is $d_k = P_k - A(i,j)$ met P_k zoals in figuur 2.5,
gem(d_k) = gemiddelde$\{d_k : \mu_Z(d_k) \neq 0\}$
Gebruik makend van deze regels wordt de definitieve output berekend, waarbij eerst de verschillende gewichten λ_m, $m = 1 \ldots 6$ berekend worden:

$$\lambda_1 = \text{med}\{\mu_{\text{NB}}(d_k) : \mu_{\text{NB}}(d_k) \neq 0\} \times \mu_{\text{more}}(\frac{\text{aantal } d_k \text{ waarvoor } \mu_{\text{NB}} \neq 0}{\text{totaal aantal } d_k})$$

$$\lambda_2 = \text{med}\{\mu_{\text{NM}}(d_k) : \mu_{\text{NM}}(d_k) \neq 0\} \times \mu_{\text{more}}(\frac{\text{aantal } d_k \text{ waarvoor } \mu_{\text{NM}} \neq 0}{\text{totaal aantal } d_k})$$



Figuur 2.27: Triangulaire lidmaatschapsfuncties

$$\lambda_3 = \text{med}\{\mu_{\text{NS}}(d_k) : \mu_{\text{NS}}(d_k) \neq 0\} \times \mu_{\text{more}}(\frac{\text{aantal } d_k \text{ waarvoor } \mu_{\text{NS}} \neq 0}{\text{totaal aantal } d_k})$$

$$\lambda_4 = \text{med}\{\mu_{\text{PS}}(d_k) : \mu_{\text{PS}}(d_k) \neq 0\} \times \mu_{\text{more}}(\frac{\text{aantal } d_k \text{ waarvoor } \mu_{\text{PS}} \neq 0}{\text{totaal aantal } d_k})$$

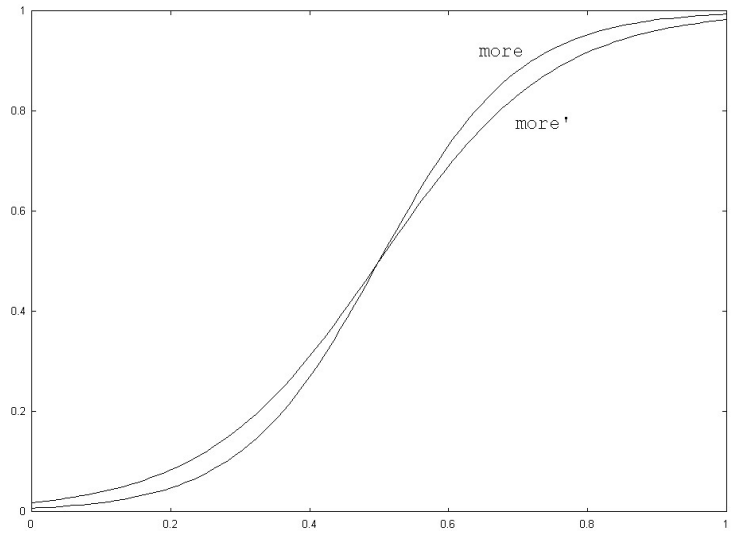
$$\lambda_5 = \text{med}\{\mu_{\text{PM}}(d_k) : \mu_{\text{PM}}(d_k) \neq 0\} \times \mu_{\text{more}}(\frac{\text{aantal } d_k \text{ waarvoor } \mu_{\text{PM}} \neq 0}{\text{totaal aantal } d_k})$$

$$\lambda_6 = \text{med}\{\mu_{\text{PB}}(d_k) : \mu_{\text{PB}}(d_k) \neq 0\} \times \mu_{\text{more}}(\frac{\text{aantal } d_k \text{ waarvoor } \mu_{\text{PB}} \neq 0}{\text{totaal aantal } d_k})$$

De correctieterm wordt dan:

$$y(i,j) = m.\text{gem}(d_k) + (1-m)\sum_{n=1}^{6} C_n\lambda_n$$

waarbij $m = \mu_{more'}(\frac{\text{aantal } d_k \text{ waarvoor } \mu_{\text{Z}} \neq 0}{\text{totaal aantal } d_k})$
en C_n is telkens het centrumpunt van de lidmaatschapsfunctie horende bij de n^{de} regel. De IFCF-filter wordt iteratief toegepast en de uitvoer van de filter wordt: $A'(i,j) = A(i,j) + y(i,j)$. Een beeld gefilterd met de IFCF-filter wordt getoond in figuur 2.29.



Figuur 2.28: Lidmaatschapsfuncties van more en more'

Figuur 2.29: IFCF-filter met 2 iteraties toegepast op Lena vervuild met zout&peper-ruis: (a)Origineel beeld; (b)$\delta = 0.05$; (c)$\delta = 0.25$; (d)$\delta = 0.50$

2.3.4 Meerstaps vaaglogisch gebaseerde filter - MFF

Idee De multipass fuzzy filter[9] werkt in drie stappen. In de eerste stap wordt met behulp van een FIRE-mechanisme de impuls-ruis weggewerkt. In de tweede stap wordt de gaussische ruis afgezwakt en in de derde stap wordt een foutcorrectie doorgevoerd.

Wegwerken van impuls-ruis Om de impuls-ruis weg te werken wordt een 3×3 invoervenster gebruikt, met de te filteren pixel als centrale pixel. 10 verschillende patronen worden onderzocht, zoals weergegeven in figuur 2.30. Noemen we weer $d_k = P_k - A(i,j)$ met P_k zoals in figuur 2.5, dan komen met de onderzochte patronen 12 verzamelingen overeen:

$$\begin{aligned} I_1 &= \{d_2, d_4, d_7\} & I_2 &= \{d_2, d_4, d_5\} \\ I_3 &= \{d_2, d_5, d_7\} & I_4 &= \{d_4, d_5, d_7\} \\ I_5 &= \{d_3, d_4, d_6, d_7\} & I_6 &= \{d_1, d_2, d_4, d_8\} \\ I_7 &= \{d_2, d_3, d_5, d_6\} & I_8 &= \{d_1, d_5, d_7, d_8\} \\ I_9 &= \{d_1, d_4, d_6, d_7, d_8\} & I_{10} &= \{d_1, d_2, d_3, d_4, d_6\} \\ I_{11} &= \{d_1, d_2, d_3, d_5, d_8\} & I_{12} &= \{d_3, d_5, d_6, d_7, d_8\} \end{aligned}$$

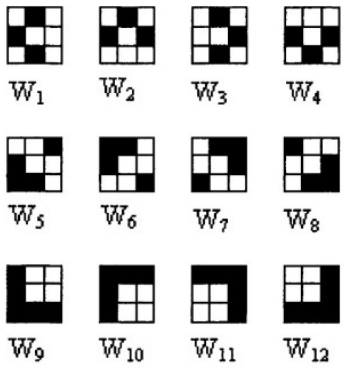
Om de correctieterm te berekenen wordt gebruik gemaakt van de vaagverzameling `large`, waarvan de grafiek van de lidmaatschapsfunctie getoond wordt in figuur 2.31. Deze wordt als volgt gedefinieerd:

$$\mu_{\mathrm{LA}}(x) = \begin{cases} 0 & -255 \leq x < a \\ \frac{b(x-a}{255(b-a)} & a \leq x < b \\ \frac{u}{255} & b \leq x \leq 255 \end{cases}$$

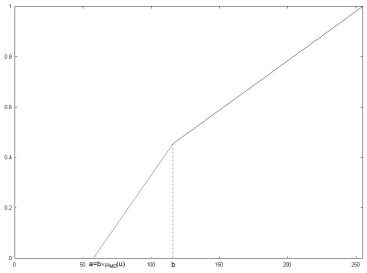
Zoals aangegeven in figuur 2.31 wordt de parameter a bepaald met behulp van een lidmaatschapsfunctie medium, die direct toegepast wordt op $A(i,j)$ en waarvan de grafiek een trapezium-vorm heeft die als volgt gedefinieerd wordt:

$$\mu_{\mathrm{MD}}(x) = \begin{cases} 0 & 0 \leq x < 128 - a_1 \\ \frac{x+a_1-128}{a_2-a_1} & 128 - a_1 \leq x < 128 - a_2 \\ 1 & 128 - a_2 \leq x < 128 + a_2 \\ \frac{x-a_1-128}{a_1-a_2} & 128 + a_2 \leq x \leq 128 + a_1 \\ 0 & 128 + a_1 \leq x \leq 255 \end{cases}$$

Typische waarden zijn: $a_1 = 100$, $a_2 = 70$ en $b = 60$.



Figuur 2.30: Patronen die onderzocht worden door de MFF-filter



Figuur 2.31: Lidmaatschapsfunctie van large

De correctieterm wordt dan:

$$\triangle r(i,j) = 255\{\max_{k=1}^{12}(\min\{\mu_{\mathrm{LA}}(d_l) : d_l \in I_k\}) - \max_{k=1}^{12}(\min\{\mu_{\mathrm{LA}}(-d_l) : d_l \in I_k\})\}$$

Deze correctieterm kan gebruikt worden om te schatten of er impulsruis aanwezig is in deze pixel en de absolute waarde $p(i,j) = |\triangle r(i,j)|$ wordt daarvoor gebruikt voor de foutcorrectie. Nu tellen we de correctieterm op bij de oorspronkelijke pixel en we krijgen:

$$r(i,j) = A(i,j) + \triangle r(i,j)$$

Na dit voor iedere pixel $A(i,j)$ gedaan te hebben, krijgen we een beeld waarin de zout&peper-ruis weggewerkt is. Dit beeld r wordt nu doorgegeven aan het tweede deel van de filter.

Verminderen van de gaussische ruis In deze stap van de filter wordt een poging gedaan om de gaussische ruis te verminderen. Hiervoor wordt gebruik gemaakt van een kruis-vormig 5×5-venster met $r(i,j)$ als centrale pixel en waarvan de pixels P_i genummerd worden van één tot en met twintig, $r(i,j)$

niet inbegrepen. We definiëren hier $d_k = P_k - r(i,j)$. Om de output van deze stap te berekenen wordt gebruik gemaakt van de vaagverzameling `small`, die een triangulaire lidmaatschapsfunctie heeft die als volgt gedefinieerd wordt:

$$\mu_{\mathrm{SM}}(x) = \begin{cases} 0 & -255 \leq x < -c \\ \frac{x+c}{2c} & -c \leq x < c \\ \frac{3c-x}{2c} & c \leq x \leq 3c \\ 0 & 3c \leq x \leq 255 \end{cases}$$

De gebruikte waarde voor de parameter c is 30. De output van de tweede stap wordt dan:

$$s(i,j) = r(i,j) + \frac{c}{20}[\sum_{k=1}^{20} \mu_{\mathrm{SM}}(d_k) - \sum_{k=1}^{20} \mu_{\mathrm{SM}}(-d_k)]$$

En op deze manier zouden we een beeld s moeten krijgen, dat bijna volledig ruis-vrij is.

Foutcorrectie Het derde blok heeft een foutcorrigerende functie en lijkt sterk op het eerste, maar maakt gebruik van de verschillen in grijswaarde tussen de 3×3-vensters in $s(i,j)$, zoals getoond in figuur 2.30 en de originele pixel $A(i,j)$. Definieer dus weer $d_k = P_k - A(i,j)$ met P_k de nummering zoals in figuur 2.5, maar dit keer is $P_k \in s$. Op deze wijze bekomen we een correctieterm $\triangle z(i,j)$:

$$\triangle z(i,j) = 255\{\max_{k=1}^{12}(\min\{\mu_{\mathrm{LA}}(d_l) : d_l \in I_k\}) - \max_{k=1}^{12}(\min\{\mu_{\mathrm{LA}}(-d_l) : d_l \in I_k\})\}$$

met de lidmaatschapsfuncties dezelfde als in de eerste stap. Om de uiteindelijke uitvoer van de filter te bepalen wordt gebruik gemaakt van de term $p(i,j)$, zoals beschreven in de eerste stap:

$$A'(i,j) = \begin{cases} A(i,j) + \triangle z(i,j) & p(i,j) > 0 \\ s(i,j) & p(i,j) = 0 \end{cases}$$

Een beeld gefilterd met de MFF-filter wordt getoond in figuur 2.32.

Figuur 2.32: MFF-filter toegepast op Lena vervuild met zout&peper-ruis en gaussische ruis: (a)Origineel beeld; (b)$\delta = 0.05$ en $\sigma = 7$; (c)$\delta = 0.25$ en $\sigma = 15$; (d)$\delta = 0.50$ en $\sigma = 25$

2.3.5 Histogram adaptieve filter - HAF

Idee De histogram adaptive filter[10] wordt gebruikt voor het verwijderen van zout&peper-ruis en zorgt ervoor dat fijne details bewaard blijven. Deze filter maakt gebruik van het histogram van het ruisbeeld om de parameters van de lidmaatschapsfuncties te bepalen. De gebruikte vaagverzamelingen zijn `dark`, `medium` en `bright`. Voor iedere vaagverzameling wordt een schatting van de uitvoer berekend en met behulp van een soort moving average filter wordt ook een algemene schatting van de uitvoer berekend. Door toepassing van enkele vaagregels wordt bepaald welke de uiteindelijke uitvoer moet worden.

Architectuur van de filter Er wordt gebruik gemaakt van een 3×3-invoervenster, waarvan de pixels P_k genummerd worden van één tot en met negen, dus ook de centrale pixel $A(i, j)$ wordt in de nummering opgenomen. De lidmaatschapsfuncties van de drie vaagverzamelingen `dark`, `medium` en `bright` zijn belvormig:

$$\mu_l(P_k) = \frac{1}{1 + (\frac{P_k - c_l}{a_l})^{2b_l}}, \quad k = 1, \ldots, 9; l = \text{DK}, \text{MD}, \text{BR} \tag{2.1}$$

De keuze van de parameters a_l, b_l en c_l wordt besproken in de volgende paragraaf. Er wordt gebruik gemaakt van de volgende regelbank:

IF $\forall P_k, k = 1, \ldots, 9$ P_k is `dark` THEN $A'(i, j)$ is `dark`
IF $\forall P_k, k = 1, \ldots, 9$ P_k is `medium` THEN $A'(i, j)$ is `medium`
IF $\forall P_k, k = 1, \ldots, 9$ P_k is `bright` THEN $A'(i, j)$ is `bright`

IF $\hat{A}(i, j)$ is closest to `dark` THEN $A'(i, j)$ is `dark`
ELSE IF $\hat{A}(i, j)$ is closest to `medium` THEN $A'(i, j)$ is `medium`
ELSE $A'(i, j)$ is `bright`

Hierin is $\hat{A}(i, j)$ de geschatte waarde van de uitvoerpixel. We definiëren nu ook nog een verzameling N_{imp} van pixels p die hoogstwaarschijnlijk ruis zijn. De pixels waarvan verondersteld wordt dat ze ruis zijn hebben een zeer grote of een zeer kleine waarde, daartoe nemen we

$$N_{\text{imp}} = \{p \in A(i, j) : p = \min(W) \vee p = \max(W) \vee p \leq T \vee p \geq (1 - T)$$

Hierbij is T een drempelwaarde die bepaalt dat alle pixels die minder dan T verschillen van de maximum- of de minimumwaarde potentiële ruispixels zijn. Hetzelfde geldt voor de maxima en minima van de 3×3 invoervensters. Om de regelbank om te zetten in een algoritme gaan we als volgt te

werk: Eerst worden de grijswaarden uit het invoervenster teruggebracht naar het interval $[0, 1]$, daarna worden deze waarden opgesplitst in 3 matrices van 3×3. In een eerste stap zijn deze 3 matrices identiek en bestaan ze gewoon uit de grijswaarden uit het invoervenster na deling door 255. In een tweede stap wordt voor iedere waarde uit de matrices de lidmaatschapsgraad berekend voor `dark`, `medium` en `bright` en deze waarden worden respectievelijk opgeslagen in de eerste, tweede en derde matrix. In een derde stap worden de waarden genormalizeerd. Elk element van iedere matrix wordt gedeeld door de som van de elementen van de matrix waartoe het te normalizeren element behoort. Uiteindelijk wordt de gewogen som van elk van de drie invoervensters berekend:

$$\text{sum}_k = \sum_{l=1}^{9} w_{lk} P_l, \quad k = DK, MD, BR, \; l = 1, \ldots, 9$$

Hierbij is P_l de naar het interval $[0, 1]$ herleide pixel uit het oorspronkelijke invoervenster en is w_{lk} de l^{de} waarde uit de matrix k. De geschatte waarde $\hat{A}(i, j)$ wordt berekend door de gemiddelde waarde te nemen van alle pixels uit het invoervenster die geen element zijn van N_{imp}:

$$\hat{A}(i,j) = \frac{\sum_{l=1}^{9} P_l}{\sum_{l=1}^{9} I_l}, \text{ waarbij } \begin{cases} P_l = 0 \text{ en } I_l = 0 & P_l \in N_{\text{imp}} \\ P_l = P_l \text{ en } I_l = 1 & P_l \notin N_{\text{imp}} \end{cases}$$

De uiteindelijke filter-output wordt toegewezen door de volgende regel:

IF $A(i, j) \in N_{\text{imp}}$
THEN $A'(i, j) = \text{sum}_k$, waarvoor $E_k = |\hat{A}(i, j) - \text{sum}_k|$ minimaal is
ELSE $A'(i, j) = \hat{A}(i, j)$

Parameters van de lidmaatschapsfuncties Door deze stap is de filter aan zijn naam gekomen. Voor het bepalen van de parameters van de lidmaatschapsfuncties wordt namelijk gebruik gemaakt van het histogram van het te filteren beeld. Zoals blijkt uit vergelijking 2.1 uit de vorige paragraaf moeten er negen parameters bepaald worden: a_k, b_k en c_k, met $k = \text{DK}, \text{MD}, \text{BR}$. Voor b_k, $k = \text{DK}, \text{MD}, \text{BR}$ is $b_k = 15$ de beste keuze gebleken. Om de andere parameters te bepalen definiëren we:

$$H[n] = \text{histogram}^3 \text{ van het beeld met ruis}$$

[3]Het histogram van een grijswaardebeeld met 255 grijswaarden is een 1×255-matrix H met $H(i)$ het aantal pixels in het beeld met grijswaarde i

$H_{\text{imp}}[n]$ =histogram van de pixels $p \in N_{\text{imp}}$
$H^e[n]$ =schatting van het histogram van het ruisvrije beeld

$H^e[n]$ wordt als volgt berekend:

$$H^e[n] = \frac{H[n] - H_{\text{imp}}[n]}{\sum_{m=0}^{255}(H[m] - H_{\text{imp}}[m])}$$

De grijswaarden worden naar het interval $[0, 1]$ herleid. Dit interval wordt verdeeld in drie gelijke delen: $[0, \frac{1}{3}]$, $[\frac{1}{3}, \frac{2}{3}]$ en $[\frac{2}{3}, 1]$ die we respectievelijk DK, MD en BR noemen. Voor ieder interval worden drie statistische variabelen berekend: pdf[4], mass en C[5]:

$$\text{pdf}_k(n) = \frac{H^e[n]}{\sum_{n\in k} H^e[n]} \quad k = \text{DK}, \text{MD}, \text{BR}$$

$$\text{mass}_k = \frac{\sum_{n\in k} H^e[n]}{\sum_{n\in[0,1]} H^e[n]} \quad k = \text{DK}, \text{MD}, \text{BR}$$

$$\text{C}_k = \sum_{n\in k} \frac{n}{255}\text{pdf}_k \quad k = \text{DK}, \text{MD}, \text{BR}$$

De waarde voor mass en C worden respectievelijk gebruikt als beginwaarden voor $a^\star$ en $c^\star$. Om optimale waarden a_k en c_k te bekomen wordt gebruik gemaakt van het volgende algoritme:
Stap 1: IF $a^\star_{\text{DK}} \geq (c^\star_{\text{DK}} - \frac{T}{255})$
THEN $a_{\text{DK}} = c_{\text{DK}} = \sqrt{c^\star_{(\text{DK}} - \frac{T}{255}) \times a^\star_{\text{DK}}}$
ELSE $a_{\text{DK}} = a^\star_{\text{DK}}$ en $c_{\text{DK}} = c^\star_{\text{DK}}$

Stap 2: IF $a^\star_{\text{BR}} \geq (1.0 - \frac{T}{255}) - c^\star_{\text{BR}}$
THEN $a_{\text{BR}} = c_{\text{BR}} = \sqrt{((1.0 - \frac{T}{255}) - c^\star_{(\text{BR}}) \times a^\star_{\text{BR}}}$
ELSE $a_{\text{BR}} = a^\star_{\text{BR}}$ en $c_{\text{BR}} = c^\star_{\text{BR}}$

Stap 3: Stel $T^\star = \frac{T}{255}$ of $(1.0 - \frac{T}{255})$
IF $a^\star_{\text{MD}} \geq |c^\star_{\text{MD}} - T^\star|$
THEN $a_{\text{MD}} = |c^\star_{\text{MD}} - T^\star|$ en $c_{\text{MD}} = c^\star_{\text{MD}}$
ELSE $a_{\text{MD}} = a^\star_{\text{MD}}$ en $c_{\text{MD}} = c^\star_{\text{MD}}$

[4]potentiële dichtheidsfunctie
[5]centröide

Hoofdstuk 3

Vergelijkende studie

3.1 Theoretisch - verschillende criteria

Om de filters op theoretisch vlak met elkaar te vergelijken, wordt gekeken naar enkele criteria. Volgende criteria komen aan bod:

- **Ruistype**: is de filter speciaal ontworpen voor gaussische ruis, werkt hij beter voor zout&peper-ruis of kan hij beide ruistypes aan?
- **Complexiteit**: hoe complex is de filter? Is er veel geheugen vereist? Hoe lang is de uitvoeringstijd?
- **Vaaggehalte**: is de filter een vaagfilter, een klassieke filter die met behulp van vaagregels verbeterd werd of een zuiver klassieke filter?
- **Iteratief/recursief**: is het de bedoeling dat de filter meerdere keren na elkaar uitgevoerd wordt? Is de filter recursief?

3.1.1 Ruistype

De onderzochte filters werden ontworpen voor impulsruis of voor gaussische ruis. Geen enkele filter werd ontworpen voor speckle ruis. De volgende klassering is gebaseerd op hetgeen de ontwerpers van de filters over hun eigen filter zeggen:

- **Impulsruis**: MF (median filter), MMF (multilevel median filter), FMF (fuzzy median filter), FDDF (fuzzy decision directed filter), WFM (weighted fuzzy mean filter), FMMF (fuzzy multilevel median filter), GMED (gaussian fuzzy filter with median center), TMED (symmetrical triangle fuzzy filter with median center), ATMED (asymmetrical

triangle fuzzy filter with median center), RR (Russo-Ramponi fuzzy filter), DS-FIRE (dual step fire-filter), HAF (histogram adaptive filter)

- **Gaussische ruis**: MAV (moving average filter), GF (gaussian filter), GMAV (gaussian fuzzy filter with moving average center), TMAV (symmetrical triangle fuzzy filter with moving average center), ATMAV (asymmetrical triangle fuzzy filter with moving average center), DW-MAV (decreasing weight fuzzy filter with moving average center)
- **Impulsruis en gaussische ruis**: IFCF (iterative fuzzy control based filter), MFF (multipass fuzzy filter)

3.1.2 Complexiteit

De complexiteit van de algoritmen hier beschreven zal onderzocht worden in tijd en in ruimte. Om de tijdscomplexiteit te bepalen, wordt gebruik gemaakt van uitvoeringstijden en om de geheugencomplexiteit te bepalen wordt de hoeveelheid geheugen bepaald die elk algoritme nodig heeft.

Tijdscomplexiteit

De rekentijden van de verschillende filters worden gegeven in tabel 3.1. Deze rekentijden zijn afhankelijk van verschillende factoren, maar ze geven wel de mogelijkheid tot classificatie van de filters. In tabel 3.1 wordt het gemiddelde van 12 meetbeurten per filter gegeven: 3 met een laag percentage zout&peper-ruis, 3 met een hoog percentage zout&peper-ruis, 3 met gaussische ruis met een kleine standaardafwijking en 3 met gaussische ruis met een hoge standaardafwijking. Een volledig overzicht van de meetresultaten kan gevonden worden in tabel B.1.
Wat opvalt in de tabel zijn de hoge uitvoeringstijden van de vaagfilters. Dit kan verklaard worden door de vele verschillende variabelen die bij de vaagfilters berekend moeten worden. Bij de MFF-filter moet rekening gehouden worden met het feit dat deze filter eigenlijk een samenstelling is van drie filters. De relatief hoge uitvoeringstijd van de GMED- en de GMAV-filter heeft dan weer te maken met het gebruik van het getal e en de machtsverheffing, die een dure bewerking is. Bij de GF-filter wordt hiervan ook gebruik gemaakt, maar slechts 1 maal, aangezien voor iedere pixel dezelfde gewichten gebruikt worden.

Tabel 3.1: Uitvoeringstijden in ms

MF	141	MAV	90
MMF	321	GF	204
FMF	647	FDDF	160
WFM	592	FMMF	386
GMED	624	TMED	233
ATMED	205	GMAV	636
TMAV	223	ATMAV	179
DWMAV1	188	DWMAV2	200
DWMAV3	203		
RR	1131	DS-FIRE	2464
IFCF	1726	MFF	3134
HAF	2052		

Geheugencomplexiteit

Om de geheugencomplexiteit te bepalen worden enkel de belangrijkste aspecten in acht genomen. Een enkele integer is verwaarloosbaar in vergelijking met een 256×256-matrix die dus 65536 keer zoveel geheugen in beslag neemt. In de bespreking stelt L het aantal grijswaarden voor en wordt gebruik gemaakt van $n \times n$ beelden omdat het niet relevant is of het beeld nu rechthoekig of vierkant is.
Het is vanzelfsprekend dat iedere filter het geheugen nodig heeft om het te filteren beeld op te slaan. Het bijkomend vereist geheugen verschilt van filter tot filter en wordt opgesomd in onderstaande lijst:

- **MF**:
  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan
  - *temp*: venstergrootte$\times$ venstergrootte

  Gebruikt geheugen: $2n^2 + c$
- **MAV**:
  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan
  - *temp*: venstergrootte$\times$ venstergrootte

  Gebruikt geheugen: $2n^2 + c$

- **MMF**:
  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan
  - *temp*: $4\times$ venstergrootte

  Gebruikt geheugen: $2n^2 + c$
- **GF**:
  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan
  - *temp*: 3×3

  Gebruikt geheugen: $2n^2 + c$
- **FMF**:
  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan
  - $n\times n$-matrix *origineel* waarin de originele matrix opgeslagen wordt, gebruikt om de lidmaatschapsfunctie te bepalen
  - $\frac{n}{10} \times \frac{n}{10}$-matrix om de waarden van de lidmaatschapsfunctie in op te slaan
  - *temp*: venstergrootte$\times$ venstergrootte

  Gebruikt geheugen: $3,01n^2 + c$
- **FDDF**:
  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan
  - *temp*: venstergrootte $\times$ venstergrootte

  Gebruikt geheugen: $2n^2 + c$
- **WFM**:
  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan
  - $n\times n$-matrix *origineel* om de originele matrix in op te slaan; wordt gebruikt om het histogram te bepalen
  - 256×1-matrix *histogram* om het histogram in op te slaan

  Gebruikt geheugen: $3n^2 + c$
- **FMMF**:

  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan
  - $4\times$ venstergrootte $\times$ venstergrootte

  Gebruikt geheugen: $2n^2 + c$

- **GMED**:
  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan
  - *temp*: venstergrootte $\times$ venstergrootte

  Gebruikt geheugen: $2n^2 + c$

- **TMED**:
  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan
  - *temp*: venstergrootte $\times$ venstergrootte

  Gebruikt geheugen: $2n^2 + c$

- **ATMED**:
  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan
  - *temp*: venstergrootte $\times$ venstergrootte

  Gebruikt geheugen: $2n^2 + c$

- **GMAV**:
  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan
  - *temp*: venstergrootte $\times$ venstergrootte

  Gebruikt geheugen: $2n^2 + c$

- **TMAV**:
  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan
  - *temp*: venstergrootte $\times$ venstergrootte

  Gebruikt geheugen: $2n^2 + c$

- **ATMAV**:
  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan
  - *temp*: venstergrootte $\times$ venstergrootte

Gebruikt geheugen: $2n^2 + c$

- **DWMAV**:
  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan
  - *temp*: venstergrootte × venstergrootte

  Gebruikt geheugen: $2n^2 + c$
- **RR**:
  - *indexes* : 48
  - *minima* : 13

  Gebruikt geheugen: $n^2 + c$
- **DS-FIRE**:
  - *window* : 32
  - *indexes* : 124
  - *minima* : 25

  Gebruikt geheugen: $n^2 + c$
- **IFCF**:
  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan

  Gebruikt geheugen: $2n^2 + c$
- **MFF**:
  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan
  - $n \times n$-matrix p om later te bepalen welke foutcorrectie moet doorgevoerd worden
  - d : 20
  - *indexes* : 48

  Gebruikt geheugen: $3n^2 + c$
- **HAF**:
  - $n \times n$-matrix *help* om gefilterde pixels in op te slaan
  - *hist* : 256

- h_imp : 256
- h_e : 256
- $dark$: 27

Gebruikt geheugen: $2n^2 + c$

We zien dat de DS-FIRE-filter en de RR-filter relatief weinig geheugen vereisen. Dit is ook logisch aangezien deze filters recursief zijn en daardoor de gefilterde waarde rechtstreeks in het ruisbeeld kan worden opgeslagen. De filters van geheugencomplexiteit $\theta(n_3)$ zijn die filters die gebruik maken van de originele afbeelding.

3.1.3 Vaaggehalte

In deze scriptie worden twee types vaaglogisch gebaseerde filters besproken: de zuivere vaagfilters en vaagfilters die gebaseerd zijn op klassieke filters. Volgende filters zijn gebaseerd op klassieke filters:

- **FMF**: Gebaseerd op de mediaanfilter
- **FDDF**: Eender welke klassieke filter wordt verfijnd met technieken uit de vaaglogica
- **WFM**: Gebaseerd op de gemiddelde waarde filter
- **FMMF**: Gebaseerd op de meerstaps mediaanfilter
- **GMED - TMED - ATMED**: Gebaseerd op de mediaanfilter
- **GMAV - TMAV -ATMAV - DWMAV**: Gebaseerd op de gemiddelde waarde filter

Deze filters kwamen aan bod in 2.2. De overige filters zijn zuivere vaagfilters, dit betekent dat de output van deze filters volledig met behulp van vaaglogica bepaald wordt. Deze filters kwamen aan bod in 2.3.

3.1.4 Iteratief/recursief

Alle filters kunnen iteratief gemaakt worden door de filters meerdere keren na elkaar uit te voeren. Hetzelfde geldt voor recursie. Een filter kan recursief gemaakt worden door de reeds gefilterde pixels te gebruiken om de output voor de nog te filteren pixels te berekenen. Er is hier maar één filter besproken waar het echt de bedoeling van de auteur is om de filter verscheidene keren na elkaar uit te voeren, met name de IFCF-filter. De RR-filter en de DS-FIRE-filter zijn filters waarvoor de auteur recursie vraagt.

Tabel 3.2: Samenvattende tabel

Filter	Ruistype	Complexiteit		Vaaggehalte	Algoritme
		Tijd	Geheugen		
MF	impuls	laag	medium	klassiek	standaard
MAV	gaussische	laag	medium	klassiek	standaard
MMF	impuls	laag	medium	klassiek	standaard
GF	gaussische	laag	medium	klassiek	standaard
FMF	impuls	medium	hoog	gemengd	standaard
FDDF	impuls	laag	medium	gemengd	standaard
WFM	impuls	medium	hoog	gemengd	standaard
FMMF	impuls	laag	medium	gemengd	standaard
GMED	impuls	medium	medium	gemengd	standaard
TMED	impuls	laag	medium	gemengd	standaard
ATMED	impuls	laag	medium	gemengd	standaard
GMAV	gaussische	medium	medium	gemengd	standaard
TMAV	gaussische	laag	medium	gemengd	standaard
ATMAV	gaussische	laag	medium	gemengd	standaard
DWMAV1	gaussische	laag	medium	gemengd	standaard
DWMAV2	gaussische	laag	medium	gemengd	standaard
DWMAV3	gaussische	laag	medium	gemengd	standaard
RR	impuls	hoog	laag	zuiver	recursief
DS-FIRE	impuls	hoog	laag	zuiver	recursief
IFCF	beide	hoog	medium	zuiver	iteratief
MFF	beide	hoog	hoog	zuiver	standaard
HAF	impuls	hoog	medium	zuiver	standaard

3.1.5 Samenvattende tabel

In tabel 3.2 staat het besluit van de theoretische vergelijking van de filters. De verschillende criteria kunnen de volgende waarden aannemen:

- Ruistype: impuls, gaussische of beide
- Complexiteit: laag, medium of hoog
- Vaaggehalte: zuiver, gemengd of klassiek
- Algoritme: recursief, iteratief of standaard

3.2 Praktisch - performantie

3.2.1 Objectief - MSE

In dit deel bekijken we enkele MSE-waarden van gefilterde beelden. De MSE(mean square error) van een bewerkt beeld wordt als volgt bepaald:

$$MSE(A') = \frac{\sum_{i=0}^{m} \sum_{j=0}^{n} |A'(i,j) - A(i,j)|^2}{mn},$$

met het origineel beeld A een $m \times n$-matrix. De MSE-waarde is een goede indicatie voor de werking van een filter.
Er bestaan verschillende beelden die vaak als testbeeld gebruikt worden. Deze beelden hebben allen verschillende eigenschappen:

- Lena: dit is een beeld met een grote complexiteit door de fijne deeltje in het haar
- Camera: dit is een beeld met een groot contrast tussen de zwarte jas en de omgeving
- Monkey: dit is een beeld waarvan de structuur van de vacht lijkt op ruis
- Pepper: dit is een beeld met grote min of meer uniforme vlakken
- Salon: dit is een beeld met weinig variatie in de grijswaarden

De 4 nog niet gebruikte beelden kunnen gevonden worden in figuur C.25. We zullen vooral gebruik maken van Lena.
De MSE-waarden na toepassing van de filters[1] kunnen gevonden worden in tabel B.2. Hierin zien we dat enkel de FMF-filter geen wijzigingen aanbrengt aan de beelden. We zien ook dat de HAF-filter problemen oplevert voor Lena en dat vooral in Monkey veel wijzigingen worden aangebracht.

Zout&peper-ruis

In de tabellen B.3 tot en met B.6 vinden we de MSE-waarden na toepassing van de filters op Lena vervuild met zout&peper-ruis. Hieruit leiden we volgende zaken af:

[1]Wanneer niet anders vermeld of geconstrueerd zullen de filters toegepast worden met venstergrootte 3×3, met 1 iteratie en met de parameters van de lidmaatschapsfuncties zoals gegeven in de filterbespreking.

- De DS-FIRE-filter presteert overal het best.
- Ook de RR-filter, die eveneens van de FIRE-familie is, scoort redelijk goed
- Voor lage ruispercentages is de MSE-waarde na toepassing van de HAF-filter groter dan de MSE-waarde van het ruisbeeld. Naarmate het ruispercentage stijgt, presteert deze filter beter.
- De ATMAV-filter evolueert van op één na slechtste filter voor lage percentages naar derde beste filter bij zeer hoge ruispercentages.
- Zowel de MMF als de FMMF scoren niet zo goed, maar dit kan wellicht verholpen worden door de venstergrootte aan te passen.
- Na 1 iteratie scoort de IFCF-filter niet zo goed.
- Van de klassieke filters scoort de MF-filter het best voor lage tot hoge (50%) ruispercentages. Voor zeer hoge ruispercentages(75%) scoort de MAV-filter beter.

Wanneer we de filters iteratief gaan toepassen, verkrijgen we de waarden in de tabellen B.7 tot en met B.10. Uit deze tabellen besluiten we dat:
Voor lage ruispercentages:

- Het iteratief toepassen van de IFCF-filter niet leidt tot betere resultaten, wat verwonderlijk is aangezien deze filter ontworpen werd om iteratief toegepast te worden.
- 2 iteraties voldoende zijn om de beste prestaties van een filter te bereiken.
- De FMF-filter met bijna de helft kan verbeterd worden door de filter 2 keer na elkaar toe te passen.

Voor hoge ruispercentages:

- Hier de IFCF-filter wel leidt tot betere resultaten na iteratief toepassen. Deze filter verbetert zelfs nog na 10 iteraties en dit leidt tot een verbetering van het resultaat t.o.v. de eerste iteratie met wel 90%.
- De DS-FIRE-filter dubbel zo goed is als de tweede beste filter.
- Ook de MFF-filter met 80% verbetert en dit reeds na 4 iteraties.

- De klassieke MF-filter ook veel verbetert door iteratief toepassen, wat hem samen met de ATMED-filter een tweede plaats oplevert.

Ons volgend experiment met zout&peper-ruis is het aanpassen van de venstergrootte. De filters waarvan de venstergrootte niet expliciet vastligt tijdens de constructie zijn: de MF, MAV, MMF, FMMF, GMED, TMED, ATMED, GMAV, TMAV, ATMAV, DWMAV1, DWMAV2 en DWMAV3 filters. De resultaten van dit experiment staan in tabellen B.11 en B.12. Bij een laag ruispercentage zien we dat vooral de ATMAV-filter en in mindere mate de MMF-filter baat heeft bij een grotere venstergrootte(5×5). Bij grotere venstergroottes nemen de MSE-waarden voor alle filters toe, behalve bij de MMF-filter.
Wanneer we kijken naar hoge ruispercentages zien we dat

- alle filters waarvan de venstergrootte kan aangepast worden, baat hebben bij een venstergrootte van 5×5 in plaats van 3×3.

- vooral de FMMF-filter met meer dan 90% verbetert als we de venstergrootte op 11×11 brengen.

Nu kunnen we deze waarnemingen gebruiken om voor elke filter apart een zo laag mogelijke MSE-waarde te bekomen. Deze resultaten staan in tabel B.13 en tabel B.14. We zullen dit doen voor een ruispercentage van 5% en een ruispercentage van 50%.
We beginnen met een ruispercentage van 5%. We bekomen volgende resultaten:

- De DS-FIRE-filter is nog steeds veruit de beste filter. Ook de andere FIRE-filter: de RR-filter scoort goed.

- Enkel de HAF-filter scoort extreem slecht.

- Zowel de MMF als de FMMF-filter scoren goed. Dit maakt de MMF-filter de beste klassieke filter.

- De filters gebaseerd op de MAV-filter scoren niet zo goed, met uitzondering van de TMAV-filter.

Voor Lena vervuild met 50% zout&peper-ruis zien we dat:

- De DS-FIRE-filter ook hier het best scoort.

- De FMMF-filter nog steeds goed scoort, terwijl de klassiek versie ervan wegzakt naar de 12de plaats.

- De klassieke mediaanfilter(MF) de beste klassieke filter wordt als derde beste filter.
- De HAF-filter veel verbeterd is in vergelijking met het lage ruispercentage.
- Ook de ATMAV-filter een lagere MSE-waarde heeft voor een sterk vervuild beeld dan na toepassing op een weinig vervuild beeld.
- De FDDF-filter enorm slecht scoort. Dit is te verklaren door het feit dat de FDDF-filter niet filtert wanneer meer dan 1 pixel uit het invoervenster vervuild is en dit bij een hoog ruispercentage enorm veel voorkomt.

We kunnen besluiten dat voor lage ruispercentages de meeste filters, met uitzondering van de GMAV, de GMED, de ATMAV en de HAF-filter, redelijk goed scoren. Voor hoge ruispercentages scoort de DS-FIRE-filter opvallend beter dan de rest. Ook nog aanvaardbaar zijn de WFM, ATMAV, HAF, ATMED en RR filters. Iteratief toegepast komen hier ook nog de TMAV, IFCF en MFF filters bij.

Gaussische ruis

Analoog aan de bespreking van zout&peper-ruis, kunnen we uit de tabellen B.15 tot en met B.18 volgende zaken afleiden:

- Voor heel weinig ruis is de MSE-waarde van de meeste gefilterde beelden groter dan die van het ruisbeeld.
- De DS-FIRE-filter, die het best presteert voor zout&peper-ruis, is één van de slechtste filters voor gaussische ruis evenals de RR-filter.
- De FMF-filter, die eigenlijk ontworpen werd voor impulsruis, scoort hier als beste filter voor weinig ruis.
- De DWMAV-filters scoren over het algemeen goed.
- Ook hier scoort de FDDF-filter niet goed voor hoge ruiswaarden en zelfs niet voor lagere ruiswaarden.
- Van de filters ontworpen voor het wegwerken van gaussische ruis, scoort de IFCF-filter het slechtst voor hoge ruiswaarden en de GF-filter voor lage ruiswaarden.

Het effect van iteratief toepassen van de filters wordt getoond in tabellen B.19 tot en met B.22.
Voor lage ruiswaarden zien we dat:

- heel wat filters baat hebben bij iteratief toepassen.
- de IFCF-filter daar ook hier niet bij is.
- de FMMF-filter 30% kan verbeterd worden.
- de meeste MSE-waarden in de buurt van 80 liggen.

Voor hoge ruiswaarden besluiten we dat:

- iteratief toepassen betere waarden oplevert voor iedere filter.
- de GMAV-filter de beste waarde oplevenert, maar ook hier liggen de meeste MSE-waarden in de buurt van eenzelfde waarde (200), behalve bij de MMF-, FDDF-, WFM-, FMMF-, RR- en DS-FIRE-filter.
- de IFCF-filter nu 70% verbetert door iteratief toepassen.

Een laatste mogelijke aanpassing is het wijzigen van de venstergrootte. De resultaten hiervan staan in tabel B.23 en tabel B.24. Hierin zien we dat voor $\sigma = 15$ enkel de MMF- en de FMMF-filter baat hebben bij een vergroting van het venster. Bij $\sigma = 50$ verbeteren alle filters. De meeste filters bereiken hun optimale waarde bij venstergrootte 5×5 of 7×7. De MMF- en de FMMF-filter bereiken nog niet hun optimale waarde bij een venstergrootte van 17×17.
We kunnen nu weer de filters iteratief gebruiken in combinatie met aanpassing van de venstergrootte. Deze resultaten staan in tabel B.25 en tabel B.26. We kunnen besluiten dat geen enkele filter zeer goed scoort voor gaussische ruis en dat de meeste filters ongeveer even goed scoren. We zien dat bij de klassieke fiters de MAV-filter best scoort voor een lage ruiswaarde en de MF-filter voor een hoge waarde. De filters ontworpen voor gaussische ruis scoren allemaal middelmatig, behalve de GF-filter.

3.2.2 Subjectief - visueel

In dit deel worden de filters getest op hun visuele performantie, aangezien er filters zijn die goed scoren voor hun MSE-waarde, maar toch minder goed ogen. De beelden die gebruikt worden voor de visuele evaluatie van de filters, staan in bijlage C.

Zout&peper-ruis

Voor zout&peper-ruis met een laag ruispercentage scoren de MF, MMF, FMMF, TMED, TMAV en de DS-FIRE-filter zeer goed. Al deze filters staan in de top 10 bij de objectieve evaluatie. Bij de overige filters merken we op:

- De RR-filter scoort betrekkelijk goed, maar er blijven heel wat witte pixels over, vooral in delen met een hoge grijswaarde.
- De FMF-filter vormt als het ware vlekken die rond eenzelfde grijswaarde liggen, wat het beeld een waterverf-effect geeft.
- Zoals reeds vermeld bespreking, filtert de FDDF-filter niet wanneer 2 of meer pixels in het invoervenster vervuild zijn. Dit geeft strepen op het beeld. Een mogelijke oplossing zou zijn om de filter recursief toe te passen.
- De HAF-filter laat zwarte vlekken achter op het beeld.

Alle andere filters maken de afbeelding wazig. De top 3 voor beelden vervuild met 5% zout&peper-ruis wordt:

1. DS-FIRE
2. FMMF
3. MMF

Voor hoge ruispercentages scoren enkel de FMMF, WFM, DS-FIRE, ATMED en de ATMAV-filter goed tot zeer goed. Ook nog aanvaardbaar zijn de MF en de FMF-filter. Alle ander filters zijn niet meer zo aantrekkelijk. Wanneer we dit resultaat vergelijken met tabel B.14 zien we dat deze 7 filters precies de 7 filters zijn die best scoren voor hun MSE-waarden. De top 3 voor beelden vervuild met 50% zout&peper-ruis wordt:

1. DS-FIRE
2. WFM
3. ATMAV

Gaussische ruis

Bij gaussische ruis is de situatie helemaal anders. Geen enkele filter kan een bevredigend visueel resultaat aanbieden. Wanneer we toch een top 3 opstellen hebben we voor $\sigma = 15$:

1. FMMF
2. MMF
3. MFF

We zien nu absoluut geen verband meer met tabel B.25 van de MSE-waarden. Voor gaussische ruis met $\sigma = 50$ hebben we:

1. MAV
2. MFF
3. GMAV

We zien dat de aangeboden vaaglogisch gebaseerde filters hier niet echt een oplossing bieden. Ook hier zien we geen verband tussen de top 3 en tabel B.26.

De evaluatie was nogal eenzijdig gebaseerd op het Lena-beeld. Misschien dat de filters voor andere types beelden anders presteren. Dit kan misschien aanleiding geven voor verder onderzoek. Als voorsmaakje worden de 3 beste filters voor 50% zout&peper-ruis al eens getest. De originele beelden staan in figuur C.25. De gefilterde beelden in figuren C.26 tot en met C.29. We kunnen inderdaad besluiten dat sommige filters beter geschikt zijn voor een bepaald soort beeld. De WFM-filter toegepast op salon bijvoorbeeld, zorgt voor zwarte vlekken op het beeld. Deze scriptie kan misschien de basis vormen voor een uitgebreider onderzoek naar de invloed van de beeldkenmerken op de performantie van de filters.

Hoofdstuk 4

Conclusie

Er werden in totaal 22 verschillende filters met elkaar vergeleken. Hiervan waren 4 filters klassieke filters, 13 filters vaag-klassiek en 5 pure vaagfilters. Eerst werd de structuur en de opbouw van de filters met elkaar vergeleken aan de hand van enkele criteria. Daarna werd de performantie van de filters zowel objectief - aan de hand van een foutwaarde - en subjectief - aan de hand van de visuele prestaties - geëvalueerd.
We kunnen besluiten dat heel wat filters goede resultaten opleveren voor het verwijderen van zout&peper-ruis. Er is weliswaar één filter die in het oog springt en dat is de DS-FIRE-filter. Deze filter is een pure vaagfilter en presteert opvallend beter dan de andere filters.
Voor gaussische ruis moeten we met minder genoegen nemen. Geen enkele filter kan een echt bevredigende prestatie neerzetten. De DS-FIRE-filter die voor zout&peper-ruis zo goed scoorde, is nu zelfs de slechtste filter. Slechts een paar filters kunnen een beeld bekomen dat ons toch een beetje voldoening schenkt.
Het is onbegonnen werk om in een thesis alle bestaande filters met elkaar te vergelijken op alle gebieden en voor allerlei beelden met verschillende kenmerken, maar er werd toch een poging gedaan om een beeld te geven van enkele filters. Ook werd een java-programma aangereikt, waarop later kan verder gebouwd worden om nog andere filters te vergelijken.

Bijlage A

Gebruikte symbolen en afkortingen

- (i, j)=positie van een pixel.
- A=het beeld.
- $A(i, j)$=grijswaarde van de pixel op positie (i, j) in het beeld A.
- A'=het gefilterd beeld.
- ATMAV=asymmetrische driehoeksvormige vaaglogische filter met gemiddelde waarde centrum.
- ATMED=asymmetrische driehoeksvormige vaaglogische filter met mediaancentrum.
- δ=ruisdichtheidsfactor van zout&peper-ruis.
- DS-FIRE=tweestaps firefilter.
- DWMAV=Afnemend gewicht vaaglogische filter met gemiddelde waarde centrum.
- FDDF=vaaglogische beslissingsgebaseerde filter.
- FMF=vaaglogische mediaanfilter.
- FMMF=vaaglogische meerstaps mediaanfilter.
- GF=gaussiche filter.
- GMAV=gaussiaanse vaaglogische filter met gemiddelde waarde centrum.

- GMED=gaussiaanse vaaglogische filter met mediaancentrum.
- HAF=histogram adaptieve filter.
- IFCF=iteratieve vaaglogische gebaseerde filter
- L=de maximum grijswaarde.
- MAV=gemiddelde waardefilter.
- MF=mediaanfilter.
- MFF=meerstaps vaaglogisch gebaseerde filter.
- MMF=meerstaps mediaanfilter.
- μ=standaardnotatie voor een willekeurige lidmaatschapsfunctie.
- med(x)=de mediaan van x.
- $m \times n$=de grootte van een matrix of beeld.
- RGB=rood-groen-blauw.
- RR=Russo-Ramponi filter
- σ=standaardafwijking bij gaussiche ruis.
- TMAV=symmetrische driehoeksvormige vaaglogische filter met gemiddelde waarde centrum.
- TMED=symmetrische driehoeksvormige vaaglogische filter met mediaancentrum.
- $W_{(i,j)}$=invoervenster met centrale pixel (i, j).
- WFM=gewogen vaag gemiddelde filter.

Bijlage B

Numerieke resultaten van de experimenten

In deze bijlagen staan de resultaten van de experimenten die gedaan werden om de filters te evalueren. De eerste tabel toont de uitvoeringstijden van de filters, daarna staan enkele resultaten van experimenten met zout&peper-ruis en uiteindelijk de resultaten van de experimenten met gaussische ruis.

Tabel B.1: Uitvoeringstijden

	5% zout&peper	5% zout&peper	5% zout&peper	50% zout&peper	50% zout&peper	50% zout&peper	gaussische ruis $\sigma = 7$	gaussische ruis $\sigma = 7$	gaussische ruis $\sigma = 7$	gaussische ruis $\sigma = 25$	gaussische ruis $\sigma = 25$	gaussische ruis $\sigma = 25$	gemiddelde
MF	140	130	210	100	101	100	111	251	111	220	110	110	141
MAV	100	90	80	80	80	80	180	100	80	70	70	70	90
MMF	421	371	291	400	400	290	290	330	290	301	291	291	321
GF	300	170	160	170	170	381	160	161	160	161	300	160	204
FMF	671	771	631	621	621	621	631	691	631	621	621	630	647
FDDF	181	170	150	150	150	160	150	171	140	160	170	161	160
WFM	721	611	541	541	551	551	561	601	661	591	581	590	592
FMMF	400	380	360	350	351	471	361	390	361	360	370	481	386
GMED	641	631	611	611	611	611	621	641	631	641	621	621	624
TMED	250	231	221	220	220	220	220	341	220	210	220	220	233
ATMED	221	310	200	191	190	190	200	210	190	191	181	181	205
GMAV	631	631	611	721	601	601	611	641	621	611	731	621	636
TMAV	340	230	210	200	210	211	210	230	210	210	210	210	223
ATMAV	200	201	180	170	171	170	171	200	171	170	170	170	179
DWMAV1	201	200	180	191	180	180	180	211	190	180	180	180	188
DWMAV2	220	210	311	180	180	180	190	200	190	181	181	181	200
DWMAV3	200	210	200	180	190	181	180	330	191	190	190	190	203
RR	1262	1111	1101	1092	1212	1121	1091	1092	1122	1102	1132	1132	1131
DS-FIRE	2674	2644	2393	2393	2413	2414	2393	2653	2393	2404	2404	2384	2464
IFCF	1733	1872	1703	1723	1723	1722	1683	1733	1663	1812	1672	1672	1726
MFF	3555	3475	3014	3044	3164	3045	3064	3405	2964	2964	2954	2964	3134
HAF	2093	2053	1903	2223	1883	1892	1893	2043	1993	2224	2213	2213	2052

Tabel B.2: MSE-waarden na toepassing van de filters op niet-vervuilde beelden

Filter	lena	camera	monkey	peppers	salon	gemiddelde
FMF	0	0	0	0	0	0
MMF	1	2	16	1	1	4
GF	113	177	311	86	150	
FDDF	1	5	28	0	2	7
DS-FIRE	1	21	16	13	9	12
FMMF	10	14	62	5	13	21
RR	3	40	46	13	5	21
IFCF	41	45	242	17	39	77
DWMAV1	45	90	228	27	45	87
MFF	55	82	257	22	54	94
DWMAV2	50	100	254	30	50	97
DWMAV3	52	104	263	31	52	100
TMAV	55	104	280	27	45	102
ATMED	44	84	322	41	44	107
MAV	56	111	284	34	56	108
MF	40	76	341	44	44	109
TMED	41	80	345	44	44	111
WFM	80	138	386	40	72	143
GMAV	111	299	277	56	62	161
GMED	107	292	324	68	60	170
ATMAV	1349	3545	300	796	461	1290
HAF	5870	245	319	116	139	1338

Tabel B.3: MSE-waarden voor de verschillende filters toegepast op Lena met 5% zout&peper-ruis

Filter				Gemiddelde
RUISBEELD	990	988	1010	996
DS-FIRE	4	5	4	4
RR	23	21	20	21
FMMF	40	32	32	35
MF	45	44	45	45
TMED	47	45	46	46
TMAV	56	57	57	57
ATMED	59	59	60	59
MFF	60	61	61	61
IFCF	65	63	62	63
FMF	74	73	83	77
WFM	82	80	81	81
GMED	92	89	86	89
GMAV	102	98	95	98
DWMAV2	177	178	180	178
DWMAV3	177	178	180	178
DWMAV1	178	179	181	179
MAV	179	181	183	181
FDDF	182	190	171	181
MMF	195	202	186	194
GF	238	235	240	238
ATMAV	999	917	967	961
HAF	3428	3484	3445	3452

Tabel B.4: MSE-waarden voor de verschillende filters toegepast op Lena met 25% zout&peper-ruis

Filter				Gemiddelde
RUISBEELD	4436	4476	4522	4478
DS-FIRE	29	25	27	27
WFM	88	89	89	89
ATMED	100	102	102	101
RR	128	118	105	117
MF	130	129	135	131
TMED	141	141	145	142
GMED	148	148	154	150
TMAV	158	157	159	158
FMF	167	153	159	160
MFF	245	258	255	253
GMAV	258	266	273	266
ATMAV	311	291	322	308
HAF	555	560	234	450
IFCF	595	627	631	618
MAV	733	755	769	752
DWMAV3	735	758	771	755
FMMF	748	749	767	755
DWMAV2	740	762	776	759
DWMAV1	762	784	799	782
GF	797	839	814	817
FDDF	2672	2730	2817	2740
MMF	2696	2756	2840	2764

Tabel B.5: MSE-waarden voor de verschillende filters toegepast op Lena met 50% zout&peper-ruis

Filter				Gemiddelde
RUISBEELD	7981	7984	7859	7941
DS-FIRE	78	74	81	78
WFM	108	109	108	108
ATMAV	208	195	204	202
HAF	265	230	277	257
ATMED	298	300	276	291
RR	311	298	314	308
GMED	775	782	718	758
TMAV	781	777	736	765
TMED	833	841	773	816
MF	853	860	780	831
FMF	882	908	795	862
GMAV	919	926	881	909
MFF	1133	1135	1091	1120
MAV	1521	1525	1486	1511
DWMAV3	1528	1533	1474	1512
DWMAV2	1537	1542	1483	1521
DWMAV1	1579	1585	1524	1563
GF	1552	1586	1601	1580
IFCF	2349	2374	2281	2335
FMMF	3109	3131	2972	3071
MMF	6631	6623	6500	6585
FDDF	6664	6648	6528	6613

Tabel B.6: MSE-waarden voor de verschillende filters toegepast op Lena met 75% zout&peper-ruis

Filter				Gemiddelde
RUISBEELD	10627	10732	10708	10689
DS-FIRE	158	174	188	173
WFM	232	220	231	228
ATMAV	294	288	283	288
HAF	472	447	384	434
RR	668	661	697	675
ATMED	776	776	805	786
GMAV	1910	1923	1966	1933
TMAV	1998	2041	2094	2044
GMED	2195	2216	2283	2231
MAV	2254	2273	2304	2277
DWMAV3	2265	2284	2315	2288
DWMAV2	2278	2297	2328	2301
GF	2307	2342	2330	2326
DWMAV1	2334	2352	2384	2357
TMED	2414	2448	2513	2458
FMF	2528	2675	2650	2618
MF	2604	2633	2713	2650
MFF	2880	2987	3066	2978
IFCF	4519	4543	4595	4552
FMMF	6097	6133	6213	6148
MMF	9702	9825	9778	9768
FDDF	9980	9989	9954	9974

Tabel B.7: MSE-waarden voor de verschillende filters toegepast met 2 iteraties op Lena met 5% zout&peper-ruis

Filter				Gemiddelde
RUISBEELD	961	981	995	979
DS-FIRE	4	5	5	5
RR	16	17	18	17
FMMF	38	33	32	34
MF	50	50	51	50
TMED	51	51	51	51
FMF	73	50	50	58
IFCF	63	63	63	63
ATMED	65	65	66	65
MFF	81	82	83	82
TMAV	86	82	85	84
WFM	104	104	105	104
MAV	143	145	147	145
DWMAV3	144	146	147	146
DWMAV1	144	146	148	146
DWMAV2	144	146	148	146
FDDF	160	184	172	172
MMF	174	197	185	185
GF	261	257	260	259
GMAV	361	410	381	384
GMED	502	527	505	511
ATMAV	1817	1855	1777	1816
HAF	3380	3551	3489	3473

Tabel B.8: MSE-waarden na toepassing van de filters die goed scoren bij iteraties toegepast op Lena met 5% zout&peper-ruis

Ruisbeeld: 990 Filter	1 iteratie	2 iteraties	3 iteraties	4 iteraties
MAV	183	149	155	165
MMF	197	197	197	197
FMF	82	49	54	58
FDDF	185	185		
DWMAV1	182	150	153	161
DWMAV2	180	150	155	165
DWMAV3	181	150	155	165
RR	22	17	17	17

Tabel B.9: MSE-waarden voor de verschillende filters toegepast met 2 iteraties op Lena met 50% zout&peper-ruis

Filter				Gemiddelde
RUISBEELD	7970	7956	7984	7970
DS-FIRE	53	52	55	53
ATMED	117	110	117	115
WFM	117	116	118	117
MF	198	186	200	195
GMED	217	204	213	211
FMF	226	205	231	221
RR	238	204	227	223
TMED	250	242	253	248
TMAV	262	249	254	255
MFF	290	278	280	283
HAF	288	286	304	293
GMAV	467	458	464	463
MAV	1149	1149	1150	1149
DWMAV3	1165	1164	1167	1165
DWMAV2	1174	1174	1176	1175
DWMAV1	1192	1192	1194	1193
IFCF	1197	1185	1216	1199
GF	1294	1292	1280	1289
ATMAV	1672	1634	1786	1697
FMMF	1727	1760	1797	1761
MMF	6625	6572	6607	6601
FDDF	6670	6607	6647	6641

Tabel B.10: MSE-waarden na toepassing van de filters die goed scoren bij 2 iteraties toegepast met meerdere iteraties op Lena met 50% zout&peper-ruis

Ruisbeeld: 7950 Aantal iteraties:	1	2	3	4	5	6	7	10	15
MF	782	200	125	110	107	108			
MAV	1483	1115	1026	996	989	993			
GF	1582	1278	1248	1272					
FMF	793	214	136	120	116	115	116		
FMMF	2982	1655	1222	1084	1031	1012	1005	1000	999
GMED	716	212	152	153					
TMED	773	254	182	165	159	157	156		
ATMED	275	116	107	113					
GMAV	883	446	346	308	292	285	282	317	
TMAV	767	270	195	173	162	163			
DWMAV1	1567	1167	1080	1040	1028	1028			
DWMAV2	1525	1167	1074	1042	1035	1041			
DWMAV3	1516	1157	1065	1033	1026	1031			
RR	339	225	217	216					
DS-FIRE	84	58	58	58					
IFCF	2353	1217	743	515	399	332	291	247	
MFF	1131	299	195	191	198				

Tabel B.11: MSE-waarden na toepassing van de filters met aanpassing van de venstergrootte toegepast op Lena met 5% zout&peper-ruis

Ruisbeeld:990 Venstergrootte	3	5	7	9
MF	45	80	112	145
MAV	177	161	188	223
MMF	177	30	28	32
FMF	80			
FDDF	163			
WFM	81			
FMMF	35	30	41	54
GMED	82	81	116	150
TMED	45	85	124	162
ATMED	59	101	139	174
GMAV	90	92	131	169
TMAV	55	93	136	178
ATMAV	914	138	169	205
DWMAV1	176	145	159	183
DWMAV2	174	149	168	195
DWMAV3	175	152	173	201
RR	21			
DS-FIRE	4			
IFCF	65			
MFF	60			
HAF	3381			

Tabel B.12: MSE-waarden na toepassing van de filters met aanpassing van de venstergrootte toegepast op Lena met 50% zout&peper-ruis

Ruisbeeld:7999 Venstergrootte	3	5	7	9	11
MF	899	144	146	172	200
MAV	1535	1062	960	939	945
MMF	6686	4472	2862	1893	1253
FMF	886	886	886	886	886
FDDF	6676	6676	6676	6676	6676
WFM	109	109	109	109	109
FMMF	3308	906	425	307	271
GMED	816	197	193	218	247
TMED	876	234	237	267	300
ATMED	332	111	142	177	211
GMAV	944	366	276	266	280
TMAV	808	258	229	248	277
ATMAV	236	131	178	224	267
DWMAV1	1593	1093	967	928	920
DWMAV2	1551	1068	956	926	923
DWMAV3	1542	1063	955	927	927
RR	296	296	296	296	296
DS-FIRE	79	79	79	79	79
IFCF	2411	2411	2411	2411	2411
MFF	1115	1115	1115	1115	1115
HAF	278	278	278	278	278

Tabel B.13: Beste MSE-waarden voor de verschillende filters toegepast op Lena vervuild met 5% zout&peper-ruis

ruisbeeld:989 Filter	MSE	ITERATIES	WINDOW
DS-FIRE	5	1	
RR	16	2	
MMF	24	2,3	5,5
FMMF	30	1,2	5,3
MF	45	1	3
TMED	45	1	3
FMF	52	2	
TMAV	56	1	3
ATMED	60	1	3
MFF	60	1	
IFCF	64	2	
WFM	81	1	
GMED	81	1	5
GMAV	92	1	5
ATMAV	138	1	5
MAV	144	2	3
DWMAV1	145	1,2	5,3
DWMAV2	145	2	3
DWMAV3	145	2	3
FDDF	162	1	
GF	236	1	
HAF	3366	1	

Tabel B.14: Beste MSE-waarden voor de verschillende filters toegepast op Lena vervuild met 50% zout&peper-ruis

ruisbeeld:7999 Filter	MSE	ITERATIES	WINDOW
DS-FIRE	56	2	
FMMF	106	4	5
MF	109	6	3
WFM	109	1	
ATMED	111	1	5
FMF	120	6	
ATMAV	131	1	5
GMED	153	2	5
TMED	155	6	3
TMAV	162	5	3
RR	197	4	
MMF	200	3,4	15,13
MFF	203	4	
GMAV	238	2	7
IFCF	254	12	
HAF	278	1	
DWMAV1	920	1	11
DWMAV2	923	1	11
DWMAV3	927	1	9
MAV	939	1,2	9,7
GF	1271	1	
FDDF	6623	10	

Tabel B.15: MSE-waarden voor de verschillende filters toegepast op Lena vervuild met gaussische ruis met $\sigma = 7$

Filter				Gemiddelde
RUISBEELD	49	49	49	49
FMF	32	32	31	32
FMMF	33	33	33	33
MMF	38	38	38	38
FDDF	49	49	49	49
GF	49	49	49	49
DS-FIRE	50	50	50	50
GMED	51	51	51	51
TMAV	51	51	51	51
IFCF	51	51	51	51
MF	52	52	52	52
DWMAV1	52	52	52	52
RR	52	52	53	52
GMAV	53	52	53	53
TMED	53	53	53	53
ATMED	54	53	53	53
DWMAV2	57	57	57	57
DWMAV3	58	58	58	58
ATMAV	59	59	59	59
MFF	61	62	61	61
MAV	62	62	62	62
WFM	101	101	100	101
HAF	122	996	990	703

Tabel B.16: MSE-waarden voor de verschillende filters toegepast op Lena vervuild met gaussische ruis met $\sigma = 15$

Filter				Gemiddelde
RUISBEELD	221	224	223	223
DWMAV1	73	73	73	73
FMF	75	74	75	75
IFCF	77	77	76	77
DWMAV2	77	77	77	77
GMAV	78	78	78	78
DWMAV3	78	78	78	78
MFF	78	78	78	78
TMAV	79	79	79	79
ATMED	81	81	81	81
ATMAV	82	82	81	82
MAV	82	82	82	82
GMED	85	85	85	85
MF	87	87	87	87
GF	97	97	98	97
TMED	90	90	90	90
FMMF	119	119	118	119
HAF	120	123	121	121
MMF	162	163	161	162
WFM	179	178	177	178
FDDF	191	193	191	192
DS-FIRE	211	213	212	212
RR	215	216	215	215

Tabel B.17: MSE-waarden voor de verschillende filters toegepast op Lena vervuild met gaussische ruis met $\sigma = 25$

Filter				Gemiddelde
RUISBEELD	602	596	603	600
MFF	115	113	115	114
DWMAV1	121	119	120	120
DWMAV2	122	121	121	121
DWMAV3	123	121	122	122
MAV	127	125	125	126
GMAV	131	130	131	131
ATMAV	133	130	132	132
ATMED	138	136	138	137
TMAV	138	137	139	138
IFCF	147	146	149	147
HAF	152	154	153	153
GMED	155	153	157	155
MF	158	157	161	159
FMF	157	158	165	160
TMED	168	166	170	168
GF	184	182	182	183
WFM	319	316	324	320
FMMF	340	336	342	339
MMF	431	426	433	430
RR	441	437	443	440
FDDF	442	439	445	442
DS-FIRE	478	472	478	476

Tabel B.18: MSE-waarden voor de verschillende filters toegepast op Lena vervuild met gaussische ruis met $\sigma = 50$

Filter				Gemiddelde
RUISBEELD	2116	2099	2109	2108
DWMAV3	316	314	318	316
MAV	317	315	319	317
DWMAV2	317	315	319	317
DWMAV1	325	324	327	325
MFF	330	322	334	329
GMAV	355	351	356	354
HAF	358	354	368	360
ATMAV	367	364	372	368
GF	375	374	371	373
ATMED	381	374	385	380
TMAV	387	381	388	385
GMED	453	444	456	451
MF	471	460	475	469
TMED	502	492	506	500
FMF	525	522	535	527
IFCF	719	702	712	711
RR	840	834	849	841
WFM	927	888	913	909
DS-FIRE	940	925	935	933
FDDF	1267	1265	1273	1268
FMMF	1287	1275	1287	1283
MMF	1546	1537	1544	1542

Tabel B.19: MSE-waarden voor de verschillende filters toegepast op Lena vervuild met gaussische ruis met $\sigma = 15$, na 2 iteraties

Filter				Gemiddelde
RUISBEELD	223	222	222	222
FMF	68	69	67	68
TMAV	75	77	76	76
GMED	76	78	77	77
GMAV	77	78	78	78
MF	77	79	77	78
IFCF	79	80	79	79
ATMED	79	81	80	80
TMED	81	83	81	82
DWMAV1	84	84	84	84
DWMAV2	86	86	86	86
DWMAV3	86	87	87	87
FMMF	87	88	87	87
MAV	87	88	88	88
ATMAV	88	91	89	89
MFF	96	97	97	97
HAF	147	143	149	146
GF	147	146	147	147
MMF	161	161	161	161
FDDF	190	191	190	190
WFM	194	196	192	194
DS-FIRE	212	212	211	212
RR	215	215	214	215

Tabel B.20: MSE-waarden na toepassing van de filters die goed scoren bij 2 iteraties toegepast met meerdere iteraties op Lena met gaussische ruis met $\sigma = 15$

Ruisbeeld: 225 Aantal iteraties:	1	2	3	4
MF	87	78	81	
MMF	164	164	164	
FMF	74	67	70	
FDDF	194	193	193	
FMMF	121	89	80	78
GMED	84	77	88	
TMED	90	82	87	
ATMED	81	80	92	
TMAV	79	76	91	
RR	216	215	215	

Tabel B.21: MSE-waarden voor de verschillende filters toegepast op Lena vervuild met gaussische ruis met $\sigma = 50$, na 2 iteraties

Filter				Gemiddelde
RUISBEELD	2101	2102	2098	2100
MFF	204	207	205	205
MAV	219	221	220	220
DWMAV3	222	224	223	223
DWMAV2	224	226	225	225
GMAV	225	230	229	228
DWMAV1	227	229	229	228
TMAV	241	247	248	245
ATMED	251	255	254	253
ATMAV	256	261	259	259
HAF	292	295	290	292
GMED	288	296	296	293
MF	304	309	309	307
FMF	318	324	325	322
IFCF	322	329	328	326
TMED	327	338	338	334
GF	333	337	333	334
WFM	700	720	724	715
RR	807	816	826	816
DS-FIRE	917	929	920	922
FMMF	930	941	949	940
FDDF	1244	1244	1243	1244
MMF	1541	1538	1536	1538

Tabel B.22: MSE-waarden na iteratieve toepassing van de filters op Lena vervuild met gaussische ruis met $\sigma = 50$
. De laagste waarde voor iedere filter staat vet gedrukt.

Ruisbeeld: 2098 Aantal iteraties:	1	2	3	4	5	6	7	10	15
MF	459	309	261	239	228	221	217	213	**212**
MAV	302	220	**208**	209	216	225	235	271	337
MMF	1537	**1536**	1536	1536	1536	1536	1536	1536	1536
GF	374	**336**	376	425	477	528	578	723	949
FMF	522	325	266	240	226	217	212	204	**202**
FDDF	1265	1243	1237	1235	1234	1234	1234	**1233**	1233
WFM	883	724	672	645	635	**632**	634	655	699
FMMF	1275	949	780	683	627	595	575	552	**545**
GMED	444	296	253	233	**224**	226	232	284	351
TMED	492	338	292	271	260	253	250	**249**	258
ATMED	374	254	229	**224**	227	233	241	271	330
GMAV	351	229	202	193	**191**	198	205	248	354
TMAV	381	248	214	202	198	**194**	199	215	253
ATMAV	364	259	**247**	261	328	402	493	843	1481
DWMAV1	324	229	211	**209**	214	222	232	266	331
DWMAV2	315	225	**211**	212	218	228	240	279	356
DWMAV3	314	223	**210**	210	217	226	237	274	343
RR	828	826	**825**	825	825	825	825	825	825
DS-FIRE	925	**920**	920	920	920	920	920	920	920
IFCF	714	328	243	219	212	**211**	213	225	247
MFF	324	205	**192**	194	202	213	224	263	332
HAF	354	290	**284**	286	290	295	301	321	362

Tabel B.23: MSE-waarden na toepassing van de filters met aanpassing van de venstergrootte toegepast op Lena vervuild met gaussische ruis met $\sigma = 15$.

Ruisbeeld:222 Venstergrootte:	3	5	7	9	11
MF	86	100	130	162	
MAV	82	117	157	198	
MMF	160	122	105	99	98
GF	98				
FMF	75				
FDDF	190				
WFM	177				
FMMF	118	93	89	92	
GMED	84	99	129	161	
TMED	89	99	130	163	
ATMED	80	107	142	179	
GMAV	77	103	138	174	
TMAV	78	102	137	174	
ATMAV	82	115	154	193	
DWMAV1	73	96	126	155	
DWMAV2	77	104	136	168	
DWMAV3	78	108	142	175	
RR	214				
DS-FIRE	211				
IFCF	77				
MFF	78				
HAF	120				

Tabel B.24: MSE-waarden na toepassing van de filters met aanpassing van de venstergrootte toegepast op Lena vervuild met gaussische ruis met $\sigma = 50$.

Ruisbeeld:2106 Venstergrootte:	3	5	7	9	11	13	15	17
MF	462	256	222	233				
MAV	314	219	224	252				
MMF	1537	1097	862	725	639	581	541	514
GF	374							
FMF	521							
FDDF	1264							
WFM	921							
FMMF	1277	849	654	556	500	460	435	423
GMED	446	246	214	226				
TMED	495	248	212	226				
ATMED	375	291	329	382				
GMAV	350	213	203	224				
TMAV	381	214	203	226				
ATMAV	362	309	357	414				
DWMAV1	323	210	201	215				
DWMAV2	314	210	207	225				
DWMAV3	313	211	210	231				
RR	833							
DS-FIRE	936							
IFCF	709							
MFF	321							
HAF	359							

Tabel B.25: Beste MSE-waarden voor de verschillende filters toegepast op Lena vervuild met gaussische ruis met $\sigma = 15$

ruisbeeld:222 Filter	MSE	ITERATIES	WINDOW
FMF	69	2	
DWMAV1	74	1	3
TMAV	76	2	3
DWMAV2	77	1	3
IFCF	77	1	
GMED	77	2	3
FMMF	77	4	3
GMAV	78	1	3
MFF	78	1	
MF	78	2	3
DWMAV3	79	1	3
ATMED	80	2	3
TMED	81	2	3
ATMAV	82	1	3
MAV	82	1	3
MMF	84	5	9
GF	98	1	
HAF	120	1	
WFM	177	1	
FDDF	189	2	
DS-FIRE	211	1	
RR	213	2	

Tabel B.26: Beste MSE-waarden voor de verschillende filters toegepast op Lena vervuild met gaussische ruis met $\sigma = 50$

ruisbeeld:2106 Filter	MSE	ITERATIES	WINDOW
TMAV	179	2	5
GMAV	181	2	5
GMED	185	3	5
MFF	187	3	
TMED	188	3	5
MF	192	3	5
FMF	199	11	
DWMAV1	201	1	7
FMMF	204	6	7
DWMAV2	205	2	5
IFCF	205	6	
DWMAV3	206	2	5
MAV	207	3	3
ATMED	223	4	3
ATMAV	246	3	3
HAF	291	3	
MMF	291	11	13
GF	336	1	
WFM	634	6	
RR	806	3	
DS-FIRE	917	3	
FDDF	1232	4	

Bijlage C

Visuele resultaten van de experimenten

Figuur C.1: Beste resultaat na filtering toegepast op Lena vervuild met 5% zout&peper-ruis: (a)Origineel beeld; (b)Ruisbeeld; (c)MF; (d)MAV.

Figuur C.2: Beste resultaat na filtering toegepast op Lena vervuild met 5% zout&peper-ruis: (a)MMF; (b)GF; (c)FMF; (d)FDDF.

Figuur C.3: Beste resultaat na filtering toegepast op Lena vervuild met 5% zout&peper-ruis: (a)WFM; (b)FMMF; (c)GMED; (d)TMED.

Figuur C.4: Beste resultaat na filtering toegepast op Lena vervuild met 5% zout&peper-ruis: (a)ATMED; (b)GMAV; (c)TMAV; (d)ATMAV.

Figuur C.5: Beste resultaat na filtering toegepast op Lena vervuild met 5% zout&peper-ruis: (a)DWMAV1; (b)DWMAV2; (c)DWMAV3; (d)RR.

Figuur C.6: Beste resultaat na filtering toegepast op Lena vervuild met 5% zout&peper-ruis: (a)DS-FIRE; (b)IFCF; (c)MFF; (d)HAF.

Figuur C.7: Beste resultaat na filtering toegepast op Lena vervuild met 50% zout&peper-ruis: (a)Origineel beeld; (b)Ruisbeeld; (c)MF; (d)MAV.

Figuur C.8: Beste resultaat na filtering toegepast op Lena vervuild met 50% zout&peper-ruis: (a)MMF; (b)GF; (c)FMF; (d)FDDF.

Figuur C.9: Beste resultaat na filtering toegepast op Lena vervuild met 50% zout&peper-ruis: (a)WFM; (b)FMMF; (c)GMED; (d)TMED.

Figuur C.10: Beste resultaat na filtering toegepast op Lena vervuild met 50% zout&peper-ruis: (a)ATMED; (b)GMAV; (c)TMAV; (d)ATMAV.

Figuur C.11: Beste resultaat na filtering toegepast op Lena vervuild met 50% zout&peper-ruis: (a)DWMAV1; (b)DWMAV2; (c)DWMAV3; (d)RR.

Figuur C.12: Beste resultaat na filtering toegepast op Lena vervuild met 50% zout&peper-ruis: (a)DS-FIRE; (b)IFCF; (c)MFF; (d)HAF.

Figuur C.13: Beste resultaat na filtering toegepast op Lena vervuild met gaussische ruis met $\sigma = 15$: (a)Origineel beeld; (b)Ruisbeeld; (c)MF; (d)MAV.

Figuur C.14: Beste resultaat na filtering toegepast op Lena vervuild met gaussische ruis met $\sigma = 15$: (a)MMF; (b)GF; (c)FMF; (d)FDDF.

Figuur C.15: Beste resultaat na filtering toegepast op Lena vervuild met gaussische ruis met $\sigma = 15$: (a)WFM; (b)FMMF; (c)GMED; (d)TMED.

Figuur C.16: Beste resultaat na filtering toegepast op Lena vervuild met gaussische ruis met $\sigma = 15$: (a)ATMED; (b)GMAV; (c)TMAV; (d)ATMAV.

Figuur C.17: Beste resultaat na filtering toegepast op Lena vervuild met gaussische ruis met $\sigma = 15$: (a)DWMAV1; (b)DWMAV2; (c)DWMAV3; (d)RR.

Figuur C.18: Beste resultaat na filtering toegepast op Lena vervuild met gaussische ruis met $\sigma = 15$: (a)DS-FIRE; (b)IFCF; (c)MFF; (d)HAF.

Figuur C.19: Beste resultaat na filtering toegepast op Lena vervuild met 50% zout&peper-ruis: (a)Origineel beeld; (b)Ruisbeeld; (c)MF; (d)MAV.

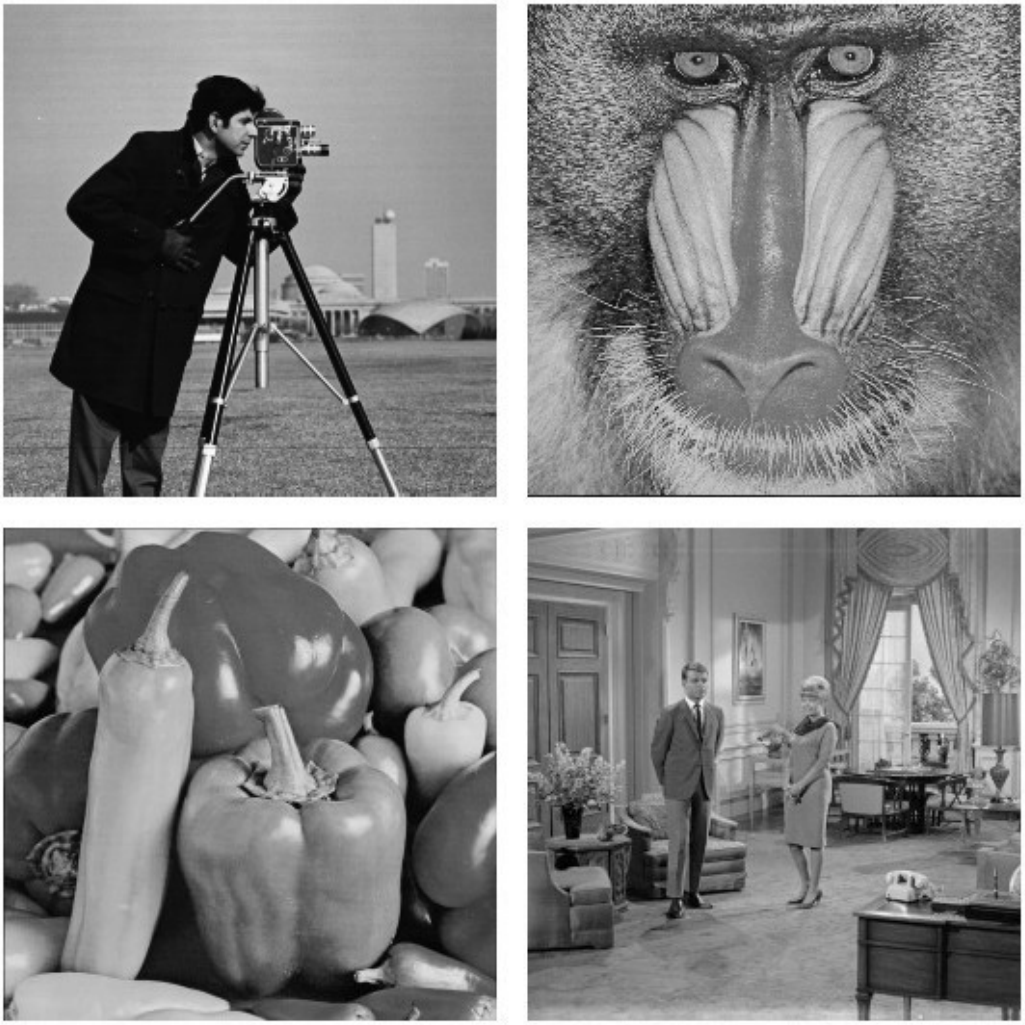
Figuur C.20: Beste resultaat na filtering toegepast op Lena vervuild met 50% zout&peper-ruis: (a)MMF; (b)GF; (c)FMF; (d)FDDF.

Figuur C.21: Beste resultaat na filtering toegepast op Lena vervuild met 50% zout&peper-ruis: (a)WFM; (b)FMMF; (c)GMED; (d)TMED.

Figuur C.22: Beste resultaat na filtering toegepast op Lena vervuild met 50% zout&peper-ruis: (a)ATMED; (b)GMAV; (c)TMAV; (d)ATMAV.

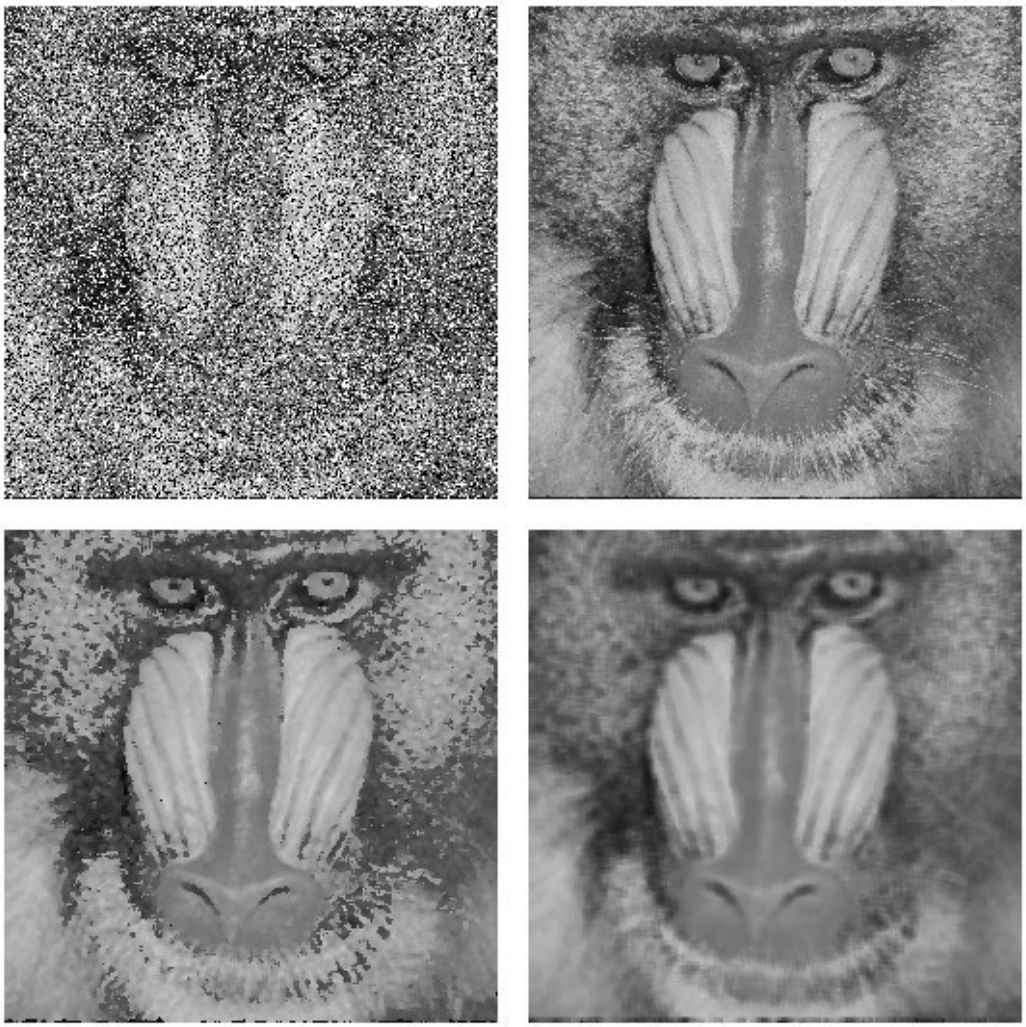
Figuur C.23: Beste resultaat na filtering toegepast op Lena vervuild met 50% zout&peper-ruis: (a)DWMAV1; (b)DWMAV2; (c)DWMAV3; (d)RR.

Figuur C.24: Beste resultaat na filtering toegepast op Lena vervuild met 50% zout&peper-ruis: (a)DS-FIRE; (b)IFCF; (c)MFF; (d)HAF.

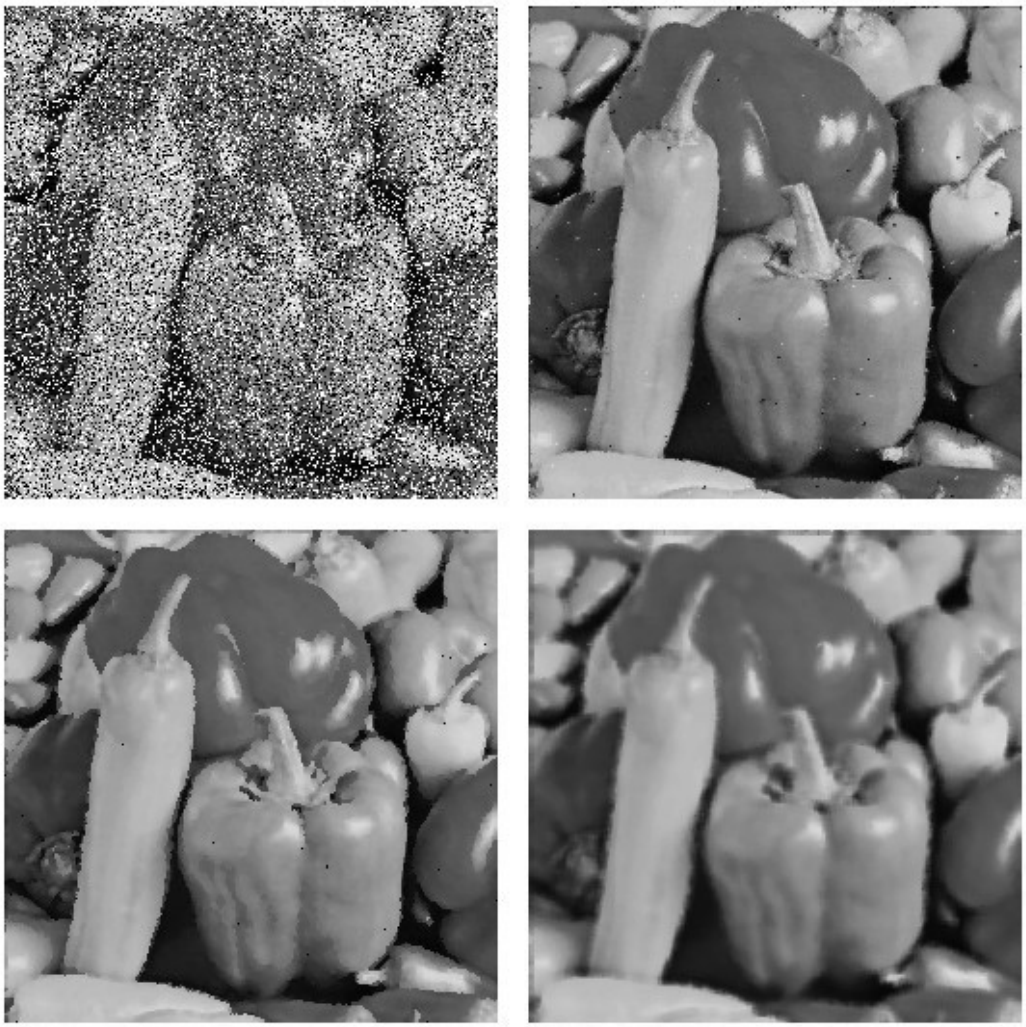


Figuur C.25: Andere afbeeldingen die worden gefilterd: (a)camera; (b)monkey; (c)pepper; (d)salon.

Figuur C.26: 3 beste filters toegepast op camera met 50% zout&peper-ruis: (a)ruisbeeld; (b)DS-FIRE; (c)WFM; (d)ATMAV.



Figuur C.27: 3 beste filters toegepast op monkey met 50% zout&peper-ruis: (a)ruisbeeld; (b)DS-FIRE; (c)WFM; (d)ATMAV.



Figuur C.28: 3 beste filters toegepast op pepper met 50% zout&peper-ruis: (a)ruisbeeld; (b)DS-FIRE; (c)WFM; (d)ATMAV.

Figuur C.29: 3 beste filters toegepast op salon met 50% zout&peper-ruis: (a)ruisbeeld; (b)DS-FIRE; (c)WFM; (d)ATMAV.

Bibliografie

[1] Kaoru Arakawa, *Median filter based on fuzzy rules and its application to image restoration*, Fuzzy sets and systems, 77(1996), 3-13

[2] M. Mancuso, R. De Luca, R. Poluzzi, G.G. Rizzotto, *A fuzzy decision directed filter for impulsive noise reduction*, Fuzzy sets and systems, 77(1996), 111-116

[3] Chang-Shing Lee, Yau-Hwang Kuo, Pao-Ta Yu, *Weighted fuzzy mean filters for image processing*, Fuzzy sets and systems, 89(1997), 157-180

[4] Mike Nachtegael, Dietrich Van der Weken, Dimitri Van De Ville, Etienne E. Kerre, *Fuzzy filters for image processing*, chapter 1, pp. 9-11 Springer Verlag, Berlin Heidelberg New York, 2003, 383p.

[5] Mike Nachtegael, Dietrich Van der Weken, Dimitri Van De Ville, Etienne E. Kerre, *Fuzzy filters for image processing*, chapter 2: fuzzy filters for noise reduction in images, pp. 28-32 Springer Verlag, Berlin Heidelberg New York, 2003, 383p.

[6] Fabrizio Russo, Giovanni Ramponi, *A fuzzy filter for images corrupted by impulse noise*, IEEE signal processing letters, vol. 3, 6(1996), 168-170

[7] Fabrizio Russo, Giovanni Ramponi, *Removal of impulse noise using a FIRE filter*, IEEE signal processing letters, 1996, 975-977

[8] Farzam Farbiz, Mohammad Bagher Menhaj, Seyed Ahmad Motamedi, *Edge preserving image filtering based on fuzzy logic*, EUFIT '98, september 7-10,1998,1417-1419

[9] Mike Nachtegael, Dietrich Van der Weken, Dimitri Van De Ville, Etienne E. Kerre, *Fuzzy filters for image processing*, chapter 1, pp. 5-9 Springer Verlag, Berlin Heidelberg New York, 2003, 383p.

[10] Jung-Hua Wang, Hsien-Chu Chiu, *HAF: an adaptive fuzzy filter for restoring highly corrupted images by histogram estimation*, Proc. Natl. Sci. Counc. ROC(A), Vol. 23, No. 5,1999, 630-643