Desenvolvimento de algoritmo para remoção da distorção causada por lente olho de peixe

Development of algorithm to remove distortion caused by fisheye lens

DOI:10.34117/bjdv7n1-447

Recebimento dos originais: 11/12/2020
Aceitação para publicação: 15/01/2021

Matheus Soares Limeira
Mestrando em Física
Instituição: Universidade Federal de Alagoas - UFAL
Endereço: Travessa Menino Marcelo, 695, Serraria, Maceió
E-mail: msoareslimeira@hotmail.com

Ygo Neto Batista
Doutor em Engenharia Elétrica
Instituição: Instituto Federal de Educação Ciência e Tecnologia de Pernambuco - IFPE
Endereço: BR 232 – Km 214 – Loteamento Redençao - Prado, Pesqueira - PE, 55200-000
E-mail: ygo@pesqueira.ifpe.edu.br

Tony Márcio Pereira da Silva
Graduando em Licenciatura em Física
Instituição: Instituto Federal de Educação Ciência e Tecnologia de Pernambuco - IFPE
Endereço: BR 232 – Km 214 – Loteamento Redençao - Prado, Pesqueira - PE, 55200-000
E-mail: tonymarcio21@gmail.com

Edivanilson Ferreira dos Santos
Graduando em Licenciatura em Física
Instituição: Instituto Federal de Educação Ciência e Tecnologia de Pernambuco - IFPE;
Endereço: BR 232 – Km 214 – Loteamento Redençao - Prado, Pesqueira - PE, 55200-000
E-mail: edivanilson1604@gmail.com

Carlos Fabiano dos Santos Silva
Graduando em Licenciatura em Física
Instituição: Instituto Federal de Educação Ciência e Tecnologia de Pernambuco - IFPE
Endereço: BR 232 – Km 214 – Loteamento Redençao - Prado, Pesqueira - PE, 55200-000
E-mail: carlosfabianos@gmail.com

Jamerson Fonseca de Sousa
Doutorando em Física
Instituição: Universidade Federal de Alagoas - UFAL
Endereço: Av. Lourival Melo Mota, S/N - Tabuleiro do Martins, Maceió - AL, 57072-900
E-mail: kdjamerson@hotmail.com

Oberlan da Silva
Mestre em Ensino de Ciencias
Instituição: Instituto Federal de Educação Ciência e Tecnologia de Pernambuco - IFPE
Endereço: BR 232 – Km 214 – Loteamento Redençao - Prado, Pesqueira - PE, 55200-000
E-mail: oberlan.silva@pesqueira.ifpe.edu.br

RESUMO
O Total Sky Imager é um equipamento que captura fotos do céu e, a partir do processamento das imagens e geralmente de redes neurais artificiais, faz previsões de curto prazo da radiação solar incidente na região. Como parte da proposta de criar um Sky Imager de baixo custo no IFPE Campus Pesqueira, foram realizados testes para aquisição das imagens através de uma lente olho de peixe acoplada a um smartphone. Contudo, essa lente introduz uma distorção indesejada na imagem adquirida, dificultando o aprendizado da rede neural artificial em etapas posteriores. O objetivo deste trabalho foi desenvolver um algoritmo capaz de remover tal distorção, baseado na física óptica da lente olho de peixe e com a aplicação de ferramentas matemáticas da geometria espacial e trigonometria. Foram estudadas opções para o algoritmo e a implementação foi em Matlab, por uma questão de disponibilidade da ferramenta, mas a implementação poderia ter sido realizada em qualquer software que implemente operações trigonométricas básicas. No algoritmo desenvolvido, a imagem é expandida (esticada) sem perdas de informações. Como a imagem final tem mais pixels do que a original, esta imagem resultante do processo contém pixels sem informações, sem prejuízos para o processamento da imagem nas etapas posteriores. Em relação ao custo computacional, basta executar o algoritmo uma vez para que as informações da distorção de uma determinada lente sejam armazenadas. A partir deste ponto, uma tabela de mapeamento de pixels pode ser implementada (look-up table) de forma a permitir a implementação do algoritmo em qualquer sistema embarcado (incluindo smartphone). Conclui-se que, com o desenvolvimento do algoritmo, é viável remover a distorção da imagem introduzida pela lente olho de peixe no desenvolvimento do protótipo do Total Sky Imager de baixo custo.

Palavras-Chave: Distorção,Lente olho de peixe, MatLab, Processamento de Imagem.

ABSTRACT
Total Sky Imager is a device that captures photos of the sky and, based on image processing and generally using artificial neural network, causes a short time to detect solar incidents in the region. As part of the proposal to create a low cost Sky Imager at IFPE Campus Pesqueira, tests were performed to capture images through a fish lens attached to a smartphone. However, this lens introduced an unwanted distortion in the image obtained, making it difficult to learn the artificial neural network in later stages. The objective of this work was to develop an algorithm capable of removing such distortion based on fisheye lens optical physics and with applications of mathematical tools of spatial geometry and trigonometry. Options for the algorithm and implementation in Matlab have been studied for the sake of tool availability, but the implementation may have been performed in any software that implements basic trigonometric operations. In the developed algorithm, an image is expanded (stretched) without information changes. Because a final image has more pixels than the original image, this unprocessed process

image contains uninformed pixels, without prejudice to image processing in later steps. Regarding the computational cost, just run the algorithm once to get stored information. From this point, a pixel mapping table can be implemented (lookup table) to allow the algorithm implementation on any embedded system (including smartphone). Conclude that, with the development of the algorithm, it is possible to remove image distortion introduced by the fisheye lens in the development of the low cost Total Sky Imager prototype.

Keywords: Distortion, Fisheye lens, MatLab, Image Processing.

1 INTRODUÇÃO

No Instituto Federal de Pernambuco, *Campus* Pesqueira, o grupo de pesquisa Interdisciplinar em Fontes Renováveis de Energia e Sistemas Eletroeletrônicos Aplicados, fundado em 2008, estuda como otimizar o rendimento de sistemas fotovoltaicos. Uma das técnicas para otimização de sistemas fotovoltaicos é realizar previsões a curto prazo da radiação incidente na região e, para realizar tal previsão, usualmente é utilizado o *Sky Imager*.

O *Sky Imager* é um equipamento que captura a imagem do céu, tentando capturar a maior área possível da abóboda celeste através de lentes específicas, e realiza o processamento desta imagem. Inicialmente a câmera deve ser configurada adequadamente, visando não saturar o sensor óptico, através do ajuste de parâmetros de exposição da câmera e, mecanicamente, ao utilizar um obstáculo para a incidência direta da radiação solar (DEV, 2014).

Em seguida, é realizado o pré-processamento da imagem, com correções de distorções originadas pela lente, ajustes de contraste e modificações nas escalas de cores. A próxima etapa é a extração de características da imagem. Nesta, informações das nuvens como quantidade, tamanho, densidade, distância em relação ao sol, são extraídas (DEV, 2014).

Por fim, a interpretação da imagem é executada através de Rede Neural Artificial (RNA) com base nas características extraídas da imagem e parâmetros utilizados, em informações geográficas, de temperatura, data e hora. Como resultado, o sistema deve utilizar a RNA para estimar a radiação solar instantânea e prever a radiação solar para curto prazo (CRISOSTO, 2019).

Portanto, o *Sky Imager* contribui para o avanço das pesquisas realizadas pelo IFPE Campus Pesqueira. Porém, está inviável adquirir um, pois, os que existem hoje no mercado são muito caros e o panorama político-econômico brasileiro inviabiliza tal

aquisição. Adicionalmente, observamos que os modelos disponíveis no mercado são volumosos e pesados, dificultando seu transporte e uso (SILVA, 2015).

O grupo de pesquisa tem desenvolvido há dois anos um *Sky Imager* próprio de baixo custo. Durante este período, cinco discentes trabalharam no desenvolvimento do equipamento, em suas diversas etapas de aquisição da imagem, pré-processamento, extração de características e interpretação.

As lentes conhecidas como olho de peixe, que são lentes bicôncavas, amplia o ângulo de abertura da lente a fim de capturar a maior área da abóbada celeste possível. Estas lentes podem também ser utilizadas em câmeras de segurança, para uso automotivo, fotografia de paisagens etc. Porém, quanto mais se afasta do centro da imagem, mais distorcida fica a imagem para o observador, fazendo com que a dimensão real dos corpos seja alterada (SCHWALBE, 2005).

O objetivo do trabalho foi desenvolver e aplicar um algoritmo no software MatLab capaz de remover a distorção de uma imagem causada por uma lente olho de peixe, especificamente a utilizada no projeto do Sky Imager. Tal distorção dificultaria o aprendizado da rede neural artificial e, consequentemente, os resultados obtidos a partir de uma base de dados restrita seriam imprecisos.

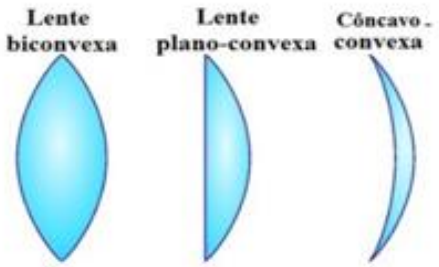
2 FUNDAMENTAÇÃO TEÓRICA

Lentes esféricas são dispositivos ópticos que formam imagens por refração da luz, de maneira que podem fazer com que os raios de luz que passam por ela convirjam ou divirjam. Existem dois tipos de lentes: as lentes convergentes e as divergentes, tendo dois pontos de foco, um para cada face, podendo ser diferentes devido a sua geometria. As lentes podem ser feitas de vidro, cristal, mas podem ser também de acrílico ou outro polímero transparente nas lentes de baixo custo (VILAS BOAS; GUALTER; BISCUOLA, 2007).

As lentes olho de peixe são chamadas também de grandes-angulares extremas, por possuírem um grande ângulo de abertura e também podem ser formadas por lentes diferentes em séries ou lentes conjugadas. As lentes grandes angulares extremas se comportam como lentes convergentes, também chamadas de bordas finas, pois os raios que passam por ela sofrem refração e convergem para um único ponto, o foco. O material de que são feitas as lentes é mais denso que o ar, isso causa um desvio da luz, e em um material homogêneo o desvio é constante. As Figuras 1 e 2 ilustram os tipos

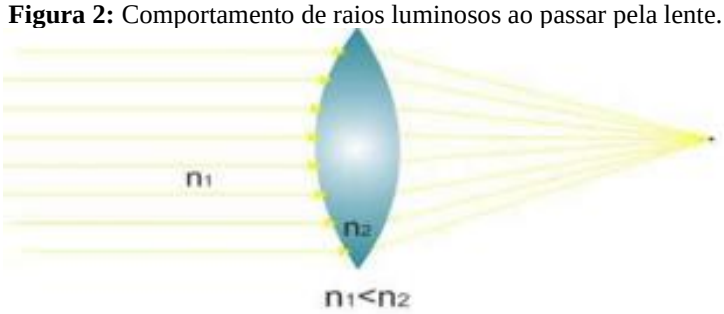
de lentes convergentes e o comportamento dessa lente e dos raios luminosos ao passar por ela (VILAS BOAS; GUALTER; BISCUOLA, 2007).

Figura 1: Tipos de lentes convergentes



Fonte: (VILAS BOAS; GUALTER; BISCUOLA, 2007

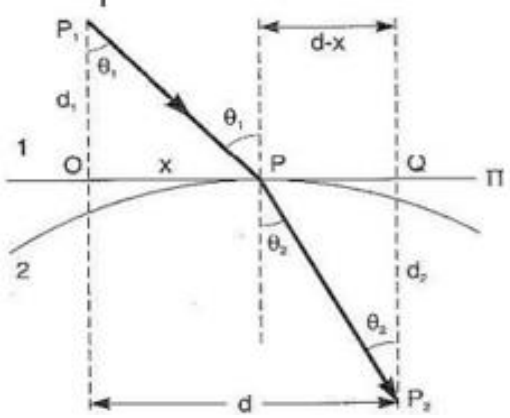
Figura 2: Comportamento de raios luminosos ao passar pela lente.



Fonte: (Só Física, 2019).

A lente delgada biconvexa possui dois raios de curvatura, cada um associado a uma face. De maneira geral, se um raio incide em uma das faces ele sofre refração ao mudar o meio de propagação, de acordo com a lei de Snell-Descartes. O Princípio de Fermat diz que de todos os caminhos para ir de um ponto a outro, a luz segue aquele que é percorrido no tempo mínimo, que corresponde a uma distância mínima, ou seja, propagação retilínea, considerando o meio homogênea. Utilizemos o modelo abaixo para mostrar de forma geral como ocorre a refração da luz ao passar do meio 1 para o 2 (NUSSENZVEIG, 1998).

Figura 3: Feixe de luz ao mudar do meio 1 para o 2.



Fonte: NUSSENZVEIG, 1998, p.12.

Utilizando a imagem acima, determinaremos o valor de x. Tomando $[P_1PP_2]$ como sendo o caminho óptico, temos

$$[P_1PP_2] = n_1.\overline{P_1P} + n_2.\overline{PP_2} = n_1.(d_1^2 + x^2)^{1/2} + n_2.[d_2^2 + (d-x)^2]^{1/2} \qquad (1)$$

Para obter o valor mínimo basta derivar em relação a x e igualar a zero:

$$0 = n_1.\frac{x}{(d_1^2 + x^2)^{1/2}} - n_2.\frac{(d-x)}{[d_2^2 + (d-x)^2]^{1/2}} = n_1.\frac{x}{\overline{P_1P}} - n_2.\frac{(d-x)}{\overline{PP_2}}$$

$$0 = n_1.\sin(\theta_1) - n_2.\sin(\theta_2) \qquad (2)$$

A primeira derivada não diz necessariamente que é um ponto de máximo ou de mínimo, o fato da primeira derivada ser zero nos diz apenas que o caminho óptico é estacionário, assim faz necessário analisar a segunda derivada para saber se de fato é o menor caminho ou se é de maior ou ponto de sela. Então quando os desvios dos caminhos são próximos, os desvios são de segunda ordem (NUSSENZVEIG, 1998). A equação 2 é a equação da Lei de Snell-Descartes

Como a lente é pequena em relação a distância para os objetos sob análise, podemos considerar que todos os raios incidentes chegam perpendicular a superfície, não sofrendo refração e convergindo para o centro óptico, que na Figura 3 é representado pelo ponto P. Ao sair da lente os raios sofrem desvios, como mostrado na lei de Snell-Descartes.

Para o nosso contexto, a lente olho de peixe permite o registro da posição dos corpos celestes em uma área maior da abóboda celeste, em relação a lentes que não são

de abertura extrema, permitindo uma análise da distribuição de nuvens com mais informações, embora distorcidas por possuir geometria esférica (SCHWALBE, 2005).

O algoritmo para remoção de distorção em lentes olho de peixe pode ser implementado em toda as linguagens de programação estudadas: C, Java, Android, etc. Contudo, o MatLab é um software que possui uma ampla biblioteca de funções matemáticas e ferramentas (*toolbox*) para processamento de imagens e implementação de redes neurais artificiais. Tais ferramentas são utilizadas no projeto, pelos outros estudantes pesquisadores, e utilizar o MatLab permite uma integração mais fácil desta etapa do projeto com as demais etapas: processamento da imagem e rede neural. O Matlab possui sua própria linguagem de programação e é bastante útil para criar sequências lógicas de comandos que possam ser repetidas e também para a criação de novas funções específicas (MARQUES; NETO, 1999).

3 METODOLOGIA

Foram desenvolvidas atividades relacionadas aos objetivos, de acordo com o exposto abaixo:

A1 – Revisão em artigos técnico-científicos:

A pesquisa foi realizada de maneira qualitativa, principalmente com auxílio do Google acadêmico e busca internacional, como o Google e o Google Imagens. Foram utilizadas as seguintes palavras-chave: “fisheye lens” e “fisheye lens distortion”. Dos resultados encontrados, foi feita uma filtragem dos conteúdos e casos que fossem relacionados diretamente com objeto de estudo, acrescentando as seguintes palavras-chave: “algorithm” e “equation”. Selecionamos três desses que tratavam diretamente da remoção da distorção de lentes do mesmo tipo, foram eles: BELLAS *et al*., 2009, SCHWALBE, 2005 e PARK; BYUN; LEE, 2009. Esses artigos encontram-se nas referências e em uma breve comparação nos resultados.

A2 - Testar o sistema óptico, com lente olho de peixe:

Uma lente olho de peixe idêntica a do protótipo criado no projeto *Sky Imager*, sob orientação do professor Prof. Ygo Batista, foi adquirida e começamos a dimensionar e a montar o cenário onde serão tiradas as fotos para realização dos testes. A figura 4 a seguir mostra a lente que foi utilizada.

Figura 4: Lente olho de peixe compacta com suporte para celular.

Fonte: (AUTORAL, 2019).

A3 – Desenvolvimento da transformada matemática para correção das distorções:

Foi realizado o estudo do MatLab e de sua linguagem para implementação das equações. As equações tiveram como base os princípios e leis da Óptica Geométrica e foram desenvolvidas utilizando a geometria do problema. Tal geometria foi estudada com o auxílio de desenhos em 2D e 3D criados no software SketchUp. As imagens utilizadas para os testes foram capturadas por integrantes do grupo de pesquisa em energias renováveis do IFPE - campus Pesqueira.

A4 – Implementação da transformada matemática no MatLab:

A transformada matemática foi colocada na linguagem de programação própria do MatLab e aplicada inicialmente em uma imagem distorcida, a fim de observar a remoção da distorção. Posteriormente o teste foi realizado com fotos tiradas de uma determinada paisagem pela câmera de um smartphone com a lente olho de peixe acoplada. Após a aplicação do algoritmo, foram feitos ajustes para obter melhores resultados.

4 RESULTADOS E DISCUSSÃO

R1 – Três artigos técnico-científicos foram selecionados e estudados.

No primeiro artigo foi encontrado uma equação que fornece um método para converter coordenadas espaciais em perspectiva de um objeto no espaço 2D (mapeamento inverso). As coordenadas 2D da imagem em formatos de matriz com i linhas e j colunas são convertidas em coordenadas no espaço 3D. Em seguida, estabelece uma função solução de terceiro grau, em que utilizou dados medidos e o método de interpolação para encontrar as constantes que serviram também como parâmetros com resultados satisfatórios (BELLAS *et al*., 2009).

O segundo estudo usou um modelo matemático para sistemas de câmeras com lentes olho de peixe e se baseou em coordenadas esféricas. O modelo foi ampliado por meio de deduções de parâmetros de distorção da lente, semelhantes ao primeiro estudo.

Os resultados foram satisfatórios, mas não apresenta as imagens finais (SCHWALBE, 2005).

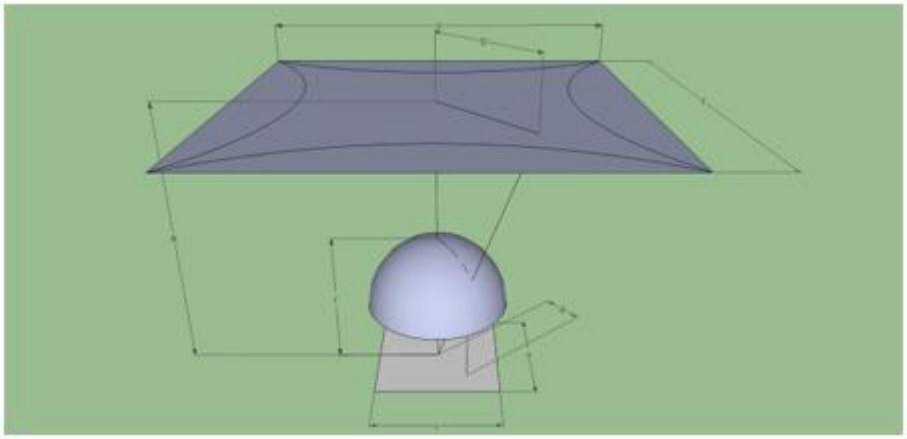
O terceiro trabalho é utilizado o modelo de distorção radial e o mapeamento inverso, onde o autor utiliza vetores de posição para a imagem utilizada distorcida e para a posição ideal da imagem não distorcida. A distorção é obtida com a subtração da posição distorcida da posição da imagem real, gerando um vetor distorção. Esse vetor distorção é dividido por coeficientes de distorção de grau dois, que é o mais comum devido a linearidade das equações, utilizando a mapeamento inverso, com uma grade 2D de pixels em posições perfeitas de linhas e colunas (i,j). Foi concluído que o método proposto é mais eficiente do que métodos mais comuns de mapeamento inverso, não precisando de aproximações (PARK; BYUN; LEE, 2009).

R2 – Lentes olho de peixe foram estudadas para entender melhor como se dá a distorção das imagens formadas, conforme brevemente apresentado na seção Referencial Teórico;

R3 – Uma lente olho de peixe compacta com suporte para celular foi adquirida e testada, e mostrou-se eficaz na sua função. Uma das imagens adquiridas é apresentada na Figura 5.

R4 – As equações foram encontradas utilizando conhecimentos de óptica geométrica e geometria espacial, com auxílio do SketchUp. Segue nas Figuras 5, 6 e 7 as indicações dos parâmetros utilizados.

Figura 5: Dimensões da imagem em perspectiva.

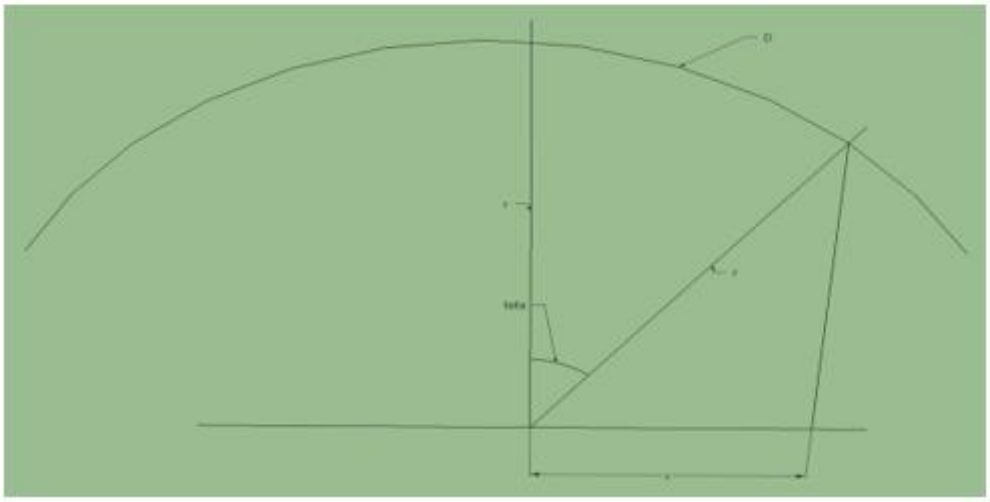


Fonte: Elaborada pelo autor.

Na Figura 5, o plano inferior (cinza claro) representa a imagem original. As dimensões x e y representam a largura e altura da imagem em pixels, respectivamente. O valor d representa a distância entre o centro da imagem e o pixel sob análise. A imagem

se molda à calota esférica na projeção dos pontos. Tal calota representa a distorção da lente. A distância entre o plano inferior e o centro da calota é chamado de “r”. O plano superior representa a imagem com a distorção removida. Os valores de X e Y são as dimensões da nova imagem. O valor de D é a do centro na nova imagem para o pixel marcado inicialmente na imagem original, ou seja, D é a projeção de d. R é a distância do centro do plano inferior para o centro do plano superior.

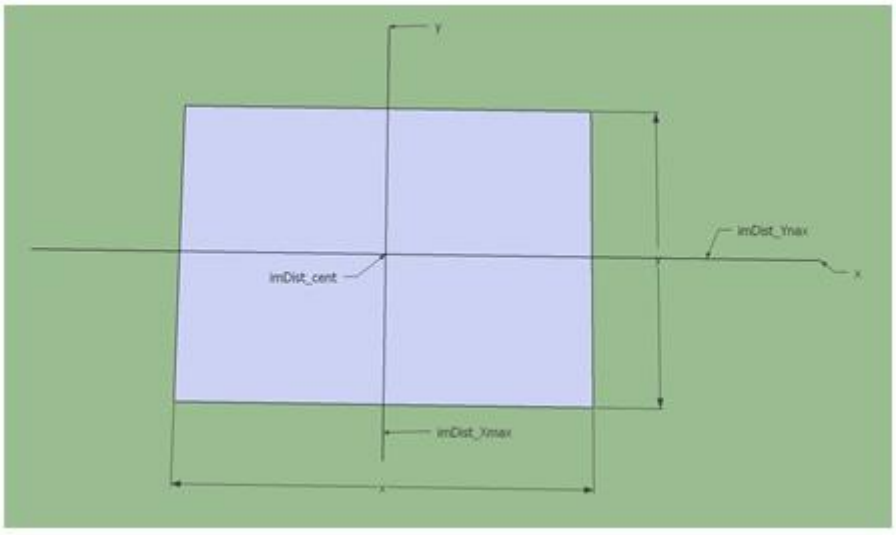
Figura 6: Geometria da imagem, lente e projeções.



Fonte: Elaborada pelo autor.

Na Figura 6, D é o comprimento do arco de circunferência, r é o raio da calota e “teta” é o ângulo de abertura do arco de circunferência, considerando zero graus o eixo azimutal. O plano inferior é a imagem original.

Figura 7: Sistemas de coordenadas da imagem original plana.



Fonte: Elaborada pelo autor.

A Figura 7 mostra o plano que representa a imagem original, onde *imDist_cent* é o centro da imagem e o *imDist_Ymáx* e *imDist_Xmáx* representam a dimensão máxima que a imagem pode ter distorcida.

As equações matemáticas encontradas inicialmente no estudo dos problemas foram obtidas modelando o plano inferior no formato da calota, deixando os vértices sem mudança de coordenadas, com o objetivo de expandir a imagem final. Depois, os pixels do plano, que agora está no formato da calota, são prologados e colocados num plano vertical, a Figura 5 mostra os três passos que foram seguidos, no plano superior, foi marcado com linha preta o formato que a imagem final tomaria. A Figura 6 mostra a relação entre ângulos de abertura da calota e distâncias entre a imagem do plano inferior. As equações obtidas foram as seguintes:

$$D(x,y) = teta(x,y).r \tag{3}$$

$$teta(x,y) = asin(\frac{\sqrt{(x - imDist_{xo})^2 + (y - imDist_{yo})^2}}{r}) \tag{4}$$

Dessa maneira observamos que o valor de r mínimo é dado por

$$r_{mín} = \sqrt{(x - imDist_{xo})^2 + (y - imDist_{yo})^2} \tag{5}$$

R5 – De acordo com os resultados R4, as equações foram implementadas no MatLab. É possível notar nas equações que há um limite para redução das distorções (rmín). Portanto, o algoritmo foi executado em 7 iterações, em um laço de repetição, conforme pode ser observado na sequência das Figuras 8 à 15.

Figura 8: Imagem original.

Fonte: AUTORAL,2019

Figura 9: Imagem original após processo 1.

Figura 10: Imagem original após processo 2.

Figura 11: Imagem original após processo 3.

Figura 12: Imagem original após processo 4.

Figura 13: Imagem original após processo 5.

Figura 14: Imagem original após processo 6.

Figura 15: Imagem original após processo 7.

Mesmo com a aplicação dos vários processos, a imagem não perdeu nenhum pixel, assim, os pontos escuros identificados na última imagem já eram esperados, pois, com o "esticamento" da imagem, a distância entre os pixels foi aumentando, criando um "vazio" entre eles.

Como o foco não foi otimização do código, considerando que a aquisição da imagem ocorrerá no *Sky Imager* em intervalos de vários minutos e que não será executado no MatLab, o tempo para o processamento da imagem, que é de aproximadamente dois minutos, não é um fator que prejudicará a aplicação do algoritmo. Adicionalmente, como o hardware será fixo, e a distorção será sempre a mesma, é possível reduzir significantemente o tempo de processamento ao fazer uma matriz de conversão fixa (*look-up table*).

Para solucionar o problema dos pontos vazios da imagem gerada pelo processo 7, foi tirada a média dos valores dos pixels vizinhos e colocado nos pontos vazios. O resultado foi o seguinte:

Figura 16: Imagem original e final.

O algoritmo também pode ser utilizado como ferramenta para o ensino em diversas áreas, principalmente nas disciplinas de graduação nas áreas tecnológicas que são mais práticas e dinâmicas e até mesmo atividades mais dinâmicas para introdução de conceitos para realização de pesquisas científicas, como nos cursos de física e engenharias em geral. É possível observar que há um melhor aproveitamento nas disciplinas que utilizam a pesquisa como forma de ensino-aprendizagem, utilizando simulações das práticas do cotidiano e para prever situações e resultados, essas estratégias didáticas-pedagógicas proporcionam ao aluno um melhor entendimento do assunto abordado, onde

esse deixa de ser apenas um observador para ser autor na construção de soluções para diversas situações reais ou simuladas (ALCÂNTARA; LIMA; DE LIMA, 2020).

5 CONCLUSÃO

O projeto começou com a ideia de retirar a distorção causada pela lente olho de peixe que existe no protótipo Total Sky Imager, equipamento em desenvolvimento no IFPE Campus Pesqueira, com o uso do software MatLab. A distorção causada pela lente olho de peixe foi removida da imagem de maneira satisfatória, obtendo bons resultados e sem perda de informações, isso facilitará a interpretação das imagem e obtenção dos dados pela rede neural do equipamento. O trabalho pode contribuir para o andamento e avanço de conhecimentos científicos e tecnológicos.

O trabalho desenvolvido deve ser implementado pelo grupo de pesquisa do IFPE no protótipo em desenvolvimento e feito melhorias para maior rendimento do equipamento, como o desenvolvimento de um algoritmo automático para geração das look-up table e a implementação do algoritmo no Android, sistema utilizado para aquisição e processamento das imagens do equipamento.

Para a criação e desenvolvimento do algoritmo foi necessário o estudo de óptica geométrica, da linguagem de programação própria do MatLab e da ferramenta de desenho computacional 3D SketchUp.

REFERÊNCIAS

ALCÂNTARA, Dalmi; LIMA, Fábio Teixeira; DE LIMA, Jonathan Gonçalves. Educação, pesquisa e recursos didáticos: Fazer educação utilizando a pesquisa como ferramenta didático-pedagógica. Brazilian Journal of Development, v. 6, n. 12, p. 95581-95595, 2020. Disponível em: https://www.brazilianjournals.com/index.php/BRJD/article/view/21183. Acesso em: 05 de janeiro de 2021.

BELLAS, Nikolaos et al. Real-time fisheye lens distortion correction using automatically generated streaming accelerators. In: 2009 17th IEEE Symposium on Field Programmable Custom Computing Machines. IEEE, 2009. p. 149-156. Disponível em: https://faculty.e-ce.uth.gr/nbellas/publications/fccm2009.pdf. Acesso em: 10 de agosto de 2018.

CRISOSTO, Cristian. Autoregressive Neural Network for Cloud Concentration Forecast from Hemispheric Sky Images. International Journal of Photoenergy, v. 2019, 2019. Disponível em:https:// www.hindawi.com/journals/ijp/2019/4375874/. Acesso em: 21 de agosto de 2019.

DEV, Soumyabrata et al. WAHRSIS: A low-cost high-resolution whole sky imager with near- infrared capabilities. In: Infrared Imaging Systems: Design, Analysis, Modeling, and Testing XXV. International Society for Optics and Photonics, 2014. p. 90711L. Disponível em: https://www.spiedigitallibrary.org/conference-proceedings-of-spie/9071/90711L/WAHRSIS--A-low- cost-high-resolution-whole-sky-imager/10.1117/12.2052982.pdf? casa_token=vjTe_fLk5_AAAAAA:d0ycD2h7DX-md5gf41vOkF_miqVs8lYDCeTLAuGXKGRfEIlpCV3XJJ_nzC9B5VFAjF0O6vGrdA. Acesso em: 10 de agosto de 2018.

MARQUES FILHO, Ogê; NETO, Hugo Vieira. Processamento digital de imagens. Brasport, 1999. Disponível em: http://projetoaprendizagemgrupo4.pbworks.com/w/file/fetch/96395952/Processamento%20Digital%20de%20Imagens.pdf. Acesso em: 19 de setembro de 2018.

VILAS BOAS, Newton; GUALTER, Ricardo Helou Doca; BISCUOLA, José. Tópicos de Física Vol. 2: Termologia, Ondulatória, Óptica. Ed. 18. São Paulo: Editora Saraiva, 2007.

NUSSENZVEIG, H. Moysés. Curso de Física Básica. Editora Edgard Bücher, v. 4, 1998.

PARK, Junhee; BYUN, Seong-Chan; LEE, Byung-Uk. Lens distortion correction using ideal image coordinates. IEEE Transactions on Consumer Electronics, v. 55, n. 3, p. 987-991, 2009. Disponível em: https://www.researchgate.net/profile/Junhee_Park4/publication/224599781_Lens_Distortion_Correction_Using_Ideal_Image_Coordinates/links/5510b6a80cf2ba84483fbf44/Lens-Distortion-Correction- Using-Ideal-Image-Coordinates.pdf. Acesso em: 10 de agosto de 2018.

SCHWALBE, Ellen. Geometric modelling and calibration of fisheye lens camera systems. Institute of Photogrammetry and Remote Sensing-Dresden University of Technology, Dresden, 2005. Disponível em: https://pdfs.semanticscholar.org/181a/e1e389c8b2748b82e6f737825a726cca5670.pdf. Acesso em: 05 de outubro de 2018.

SILVA, Rutelly Marques da. Energia solar no Brasil: dos incentivos ao desafios. 2015. Disponível em: https://www2.senado.leg.br/bdsf/bitstream/handle/id/507212/TD166-RutellyMSilva.pdf? sequence=1. Acesso em: 18 de maio de 2018.

Só Física. Lentes convergentes. Virtuous Tecnologia da Informação, 2019. Consultado em 22/11/2019 às 15:03. Disponível em: http://www.sofisica.com.br/conteudos/Otica/Lentesesfericas/convergentes.php. Acesso em: 21 de novembro de 2019.
n)

APÊNDICE

Algoritmo para correção da distorção:

```
function imFinal2 = corrigeDistorcao(imDistorcida,r)
%% PROGRAMA PARA CORRIGIR DISTORÇÕES GERADAS A PARTIR DE LENTES TIPO OLHO DE PEIXE
% Autores: Matheus Soares Limeira e Ygo Neto Batista
% 2019
% clear all; clc; %limpa todas as variáveis e o prompt de comando
%% Etapa 1: Leitura de imagem de teste distorcida
imDistorcida = im2bw(imread('tst7.jpg'),0.5);
if (mod(size(imDistorcida,2),2)==1)
    imDistorcida = imDistorcida(:,1:size(imDistorcida,2)-1,:);
end
if (mod(size(imDistorcida,2),2)==1)
    imDistorcida = imDistorcida(:,1:size(imDistorcida,2)-1,:);
end
% Limites e centro da imagem (X = colunas, Y = linhas):
imDist_xmax = size(imDistorcida,2);
imDist_ymax = size(imDistorcida,1);
imDist_xo = (1+imDist_xmax)/2;
imDist_yo = (1+imDist_ymax)/2;
% figure;
% imshow (imDistorcida);
% %% Etapa 2: Define parâmetros da distorção
r = (sqrt((x-imDist_xo)^2+(y-imDist_yo)^2); % pixels. Raio da lente olho de peixe*.
Quanto maior, menos distorcido.
%% Etapa 3: Verifica os ângulos entre a reta normal que passa no centro da imagem e
a reta que passa na projeção da imagem distorcida na calota
teta=zeros(imDist_xmax,imDist_ymax);
for x = 1:imDist_xmax
    for y = 1:imDist_ymax
        teta(x,y) = asin (sqrt((x-imDist_xo)^2+(y-imDist_yo)^2) / r);
    end
end
%% Etapa 4: Calcula o comprimento entre o ponto central da superfície da calota e o
ponto projetado da imagem original
D = zeros(imDist_xmax,imDist_ymax);
for x = 1:imDist_xmax
    for y = 1:imDist_ymax
        D(x,y) = teta(x,y)*r;
    end
```

```
end
%% Etapa 5: Criar matriz para a imagem não distorcida
alfa_diag = atan(imDist_ymax/imDist_xmax);
imFinal_xmax = ceil(max(max(D))*cos(alfa_diag))*2;
imFinal_ymax = ceil(max(max(D))*sin(alfa_diag))*2;
imFinal_xo = (1+imFinal_xmax)/2;
imFinal_yo = (1+imFinal_ymax)/2;
imFinal = uint8 (zeros (imFinal_ymax,imFinal_xmax,3));
%% Etapa 6: Criar matriz alfa da matriz original
alfa = zeros(imDist_xmax,imDist_ymax);
for x = 1:imDist_xmax
    for y = 1:imDist_ymax
        if (x>imDist_xo)
             alfa(x,y) = atan(-(y-imDist_yo) / (x-imDist_xo));
        else
             alfa(x,y) = atan((y-imDist_yo) / (x-imDist_xo));
        end
    end
end
%% Etapa 7: Mapear as coordenadas dos pixels originais na matriz final
mapeamento_x = zeros(imDist_xmax,imDist_ymax);
  for x = 1:imDist_xmax
      for y = 1:imDist_ymax
          if (x>imDist_xo)
              mapeamento_x(x,y) = D(x,y)*cos(alfa(x,y));
              mapeamento_y(x,y) = D(x,y)*sin(alfa(x,y));
          else
               mapeamento_x(x,y) = -D(x,y)*cos(alfa(x,y));
               mapeamento_y(x,y) = D(x,y)*sin(alfa(x,y));
          end
      end
  end
  %% Etapa 8: Copiar pixels da imagem original para a imagem final, na posição
  mapeada
  % E cria a matriz de pixels sem informações
  inform=boolean(zeros(size(imFinal,1),size(imFinal,2)));
  for x = 1:imDist_xmax
      for y = 1:imDist_ymax
          for z = 1:3
```

```
        imFinal(round(imFinal_yo-mapeamento_y(x,y)),round(mapeamento_x(x,y)
+imFinal_xo),z) = imDistorcida(y,x,z);
         end
         inform(round(imFinal_yo-mapeamento_y(x,y)),round(mapeamento_x(x,y)
+imFinal_xo)) = 1;
    end
end
%%
imFinal2 = imFinal;
for  x  =    3:(imFinal_xmax-3)
    for y = 3:(imFinal_ymax-3)
            if inform(y,x) == 0
                imFinal2(y,x,1) = median(median(imFinal(y-2:y+2,x-2:x+2,1)));
                imFinal2(y,x,2) = median(median(imFinal(y-2:y+2,x-2:x+2,2)));
                imFinal2(y,x,3) = median(median(imFinal(y-2:y+2,x-2:x+2,3)));
            end
    end
end
%%
% figure (2);
%     imshow(imFinal);
```

Algoritmo para aplicação dos processos:

```
imDistorcida = imread('tst7.jpg');
tic;
%%
imPasso1 = corrigeDistorcao(imDistorcida,3000);
lmax = size(imPasso1,1);
cmax = size(imPasso1,2);
imPasso1 = imPasso1(150:lmax-150,500:cmax-500,:);
toc;
%%
imPasso2 = corrigeDistorcao(imPasso1,3000);
lmax = size(imPasso2,1);
cmax = size(imPasso2,2);
imPasso2 = imPasso2(150:lmax-150,200:cmax-200,:);
toc;
%%
imPasso3 = corrigeDistorcao(imPasso2,3000);
lmax = size(imPasso3,1);
```

```
cmax = size(imPasso3,2);
imPasso3 = imPasso3(550:lmax-550,600:cmax-600,:);
toc;
%%
imPasso4 = corrigeDistorcao(imPasso3,3400);
lmax = size(imPasso4,1);
cmax = size(imPasso4,2);
imPasso4 = imPasso4(500:lmax-500,600:cmax-600,:);
toc;
%%
imPasso5 = corrigeDistorcao(imPasso4,3800);
lmax = size(imPasso5,1);
cmax = size(imPasso5,2);
imPasso5 = imPasso5(450:lmax-450,600:cmax-600,:);
toc;
%%
imPasso6 = corrigeDistorcao(imPasso5,5000);
lmax = size(imPasso6,1);
cmax = size(imPasso6,2);
imPasso6 = imPasso6(350:lmax-350,400:cmax-400,:);
toc;
%%
imPasso7 = corrigeDistorcao(imPasso6,5000);
lmax = size(imPasso7,1);
cmax = size(imPasso7,2);
imPasso7 = imPasso7(350:lmax-350,400:cmax-400,:);
toc;
%%
imshow(imPasso7);
```