Simulação de experimentos de óptica para ensino de física - Lentes

Relatório Parcial do Projeto de F 809 (Intrumentação para o ensino)

Aluno: Marcelo de Freitas Rigon **RA:** 002095
Orientador: Pedro Miguel Raggio Santos

Introdução:

Esse projeto tem o intuito de desenvolver um programa de ensino de óptica, mais especificamente lentes e fendas. Para isso, utilizaremos recursos visuais para ilustrar as várias fórmulas ou contas que o aluno poderia fazer. Isso permitirá o desenvolvimento da "intuição física" do aluno para esses efeitos, pois ele não estará lidando com um grupo de fórmulas a decorar e sim vendo uma simulação do que acontece na realidade, podendo trocar os elementos e/ou parâmetros a seu bel prazer sem ter que se preocupar com o trabalho de recalcular os resultados para esse novo conjunto de condições.

Para que o desenvolvimento desse programa fosse possível dentro do período disponível para a disciplina, utilizamos a linguagem Python com os pacotes PyGame e SciPy (endereços para informações ou cópia dos programa estão na referência e Anexo I). Além disso, o Python nos permitiu construir um programa compatível com Linux e Unix com Gtk/Motif, Mac/OS e Windows, dos que eu conheço. No entanto, só será possível verificarmos essa compatibilidade no Linux ou Windows por serem os sistemas operacionais aos quais temos acesso.

Sobre a internet e os programas:

Introdução

Atualmente, grande parte da população tem acesso fácil a um computador. Sendo assim, é, no mínimo, interessante utilizarmos esse recurso para facilitar a realização de nossas tarefas diárias.

A evolução computacional no ramo da educação é um assunto importante não só para o ensino mais fácil e rápido do conhecimento, mas também para o armazenamento de forma mais compacta e interativa de informações na área à qual o programa diz respeito.

Os programas, pelo menos na parte educacional da àrea de exatas, tratam o ensino através da óptica da simulação de experimentos diferentes com diferentes condições, de forma a tornar mais intuitivo a compreensão dos fenômenos. Esses mesmos programas podem ser utilizados para prevenir gastos desnecessários com montagens que não teriam nenhuma utilidade, seja por serem inviáveis ou pelos resultados não corresponderem ao que o interessado pretendia demonstrar. Além de ser uma forma portátil e de fácil manipulação de fazer as mais diversas demonstrações em sala de aula, por maior que seja a falta de recurso ou espaço para a aquisição dos aparatos necessários para a montagem da experiência.

Definição de "hardware" e "software"[1] ou programa

Tudo o que existe num computador pode ser classificado em duas cartegorias: o "hardware" e o "software". Se separarmos essas palavras em duas partes temos "ware" como parte comum, e "hard" que significa duro (no nosso caso) e "soft" que significa mole.

Portanto, podemos concluir que o "hardware" é a parte dura do computador. De certa forma isso é correto, mas prefiro dizer que o "hardware" é aquilo que você vê mesmo quando o computador está desligado. Por exemplo, o monitor, o gabinete (CPU ou Unidade de Processamento Central), o teclado, placa de som e vídeo. E o "software" é a parte maleável. Os programas são o "software" e são aquilo que você só vê quando o computador está ligado. Digo que o "software" ou programa é a parte maleável, pois você pode, muito facilmente, trocar os programas do seu computador e, poderia, modificar os programas para se adaptarem às suas necessidades.

Programa Livre

Os programas mais divulgados na população, atualmente, são programas proprietários. Os programas proprietários são distribuídos de forma a permitir que o usuário utilize o programa sem poder modificar e adaptá-lo ou melhorá-lo. Isso é feito não transmitindo ao usuário o código fonte (arquivo utilizado para a criação do programa) e/ou através de licenças que o proibem de modificar o programa de qualquer forma.

Os programas livres, ao contrário, incentivam a modificação do seu código fonte pelos seus usuários, de forma a evoluir, corrigir problemas ou adaptar esse programa aos mais diversos tipos de necessidades de seus usuários.

Fazendo uma analogia com roupas:

Os programas proprietários seriam como uma roupa que você compra e não pode fazer a barra, apertar a cintura e coberta por um plástico opaco que não permita que você veja como é a costura dela.

Os programas livres são as roupas da maneira que são comercializadas atualmente. Você pode apertar a cintura se a calça estiver caindo, ou fazer a barra se estiver muito comprida para que ela se torne algo nas medidas certas para você.

Uma vantagem para o programa livre: A Internet

Algo vital para o bom fucionamento de um programa livre é a existência de uma forma de troca de arquivo, idéias e informações entre os desenvolvedores do programa. Se isso não existe, acabam existindo várias versões desconexas e nem todas os melhoramentos são agrupados em um só programa. De fato, pessoas que tentavam utilizar programas nessa época acabavam tendo problemas por que o que faziam funcionava no seu computador mas não funcionava no do vizinho, pois era o mesmo número de versão mas proveniente de diferentes programadores.

Com o surgimento da internet, esse problema foi reduzido para um número muito pequeno de casos, além de se tornar uma fonte importante de informações na sociedade. A maioria dos computadores possuem alguma forma de se conectar à internet. E muitas pessoas que não tem acesso à internet usam os "cybercafés" ou outros meios, como a faculdade em que estuda, para verificar seu correio eletrônico ou procurar informações.

Dessa forma, tivemos o grande avanço do Linux que se torna cada vez mais popular, além de ter cada vez mais programas feitos para ele para substituir as ferramentas criadas para o seu principal rival proprietário, o Windows.

---

1 Utilizo aqui a nomeclatura do inglês, simplesmente, por facilitar a minha explicação.

Um impecílio para o programa livre

Apesar desse grande auxílio que foi a invenção e popularização da internet. No entanto, enquanto as pessoas não souberem programar, isso não tem tanta importância assim. Imagine de que adianta poder fazer a barra da calça se só poucas pessoas são capazes de fazer isso.

Um bom exemplo disso é uma diferença entre Brasil e Japão. No Japão você encontra muitos aparelhos eletrônicos no lixo mesmo por defeitos facilmente reparáveis. Pois no Japão são poucas as pessoas que se ocupam com reparos desses aparelhos, se tornando um serviço caro, sendo assim mais fácil comprar um produto novo e jogar esse for a. Por outro lado, no Brasil, temos muitas pessoas prestando assistência técnica para reparos de produtos eletrônicos, tornando esse problema mais fácil de ser solucionado.

Utilizando como exemplo, programas de modelagem tridimensional, por que você utilizaria o Blender que é um programa livre se você não sabe melhorá-lo e existe o Maya ou o 3d Studio que já faz um bando de coisa? Tudo bem que esses programas proprietários citados custam em torno de US$2.000,00, mas pelo menos eles serão capazes de solucionar o seu problema, e o Blender não por que programação não é algo que qualquer um sabe.

Nasce uma nova linguagem: Python

No inicio dos anos 80, Guido van Rossum trabalhava em uma equipe que estava desenvolvendo uma linguagem chamda ABC no Instituto Nacional de Pesquisa em Matemática e Computação Científica (Centrum voor Wikunde en Informatica, CWI) da Holanda. ABC era uma linguagem clara que tinha a intenção de ensinar programação a pessoas inteligentes que utilizavam computadores, mesmo não sendo programadores - seus estudos incluiam de físicos e cientistas sociais, a linguistas – mas essas pessoas ficaram surpresas com as limitações, restrições e regras arbitrárias que as linguagens estabeleciam. Entre muitas razões e a não existência de uma internet para fazer a distribuição dessa linguagem, o projeto não foi um sucesso.

Em 1986, ele se mudou para um projeto diferente no CWI, o projeto Amoeba (Amoeba era um sistema operacional da época). No final dos anos 80, eles descobriram que estavam precisando de uma linguagem de scripting. Como Guido van Rossum já tivera uma experiência com a linguagem ABC e liberdade o suficiente para criar um mini projeto dentro do escopo do seu atual trabalho, ele decidiu desenvolver uma linguagem de scripting simples que possuísse as melhores qualidades do ABC sem os seus problemas.

Pegando os ingredientes do ABC e mexendo um pouco, surgiu o Python. O Python é muito semelhante ao ABC, mas também tem algumas diferenças. Entre elas as variáveis do tipo lista e dicionário, comandos básicos, o uso de indentação para gerenciar inicio e fim de blocos de comandos. A mais importante e inovativa diferença do Python para o ABC que garante grande parte do sucesso que essa linguagem vem tendo é a facilidade para extendê-la.

Guido já sabia que ele iria querer que o Python funcionasse em diferentes plataformas. Pois ele teria que funcionar no Amoeba que eles estavam trabalhando e no UNIX que eles tinham nos seus computadores. Além disso, também deveria operar em Windows e Macintosh. Todos esses sistemas tinham características que eram iguais, como a biblioteca padrão de entrada e saída do C, mas também haviam outras funcionalidades.

Outra características importante do Python adicionada por Guido foi a possibilidade de se escrever um módulo diretamente em C e importá-lo como se estivesse em Python e os tipos de variáveis serão convertidas sem problemas e maiores cuidados. Essa idéia foi logo vista como muito boa, pois todos os seus colegas de trabalho no CWI, os usuários e ele mesmo começaram prontamente a escrever modulos de extensão que permitiam fazer

todo tipo de coisa.

Um projeto de programação para todos

Computer Programming for Everybody (CP4E) ou Programação de Computadores para todos, é um projeto desenvolvido pelo Guido van Rossum que afirma ser tão importante para o homem atual o domínio mínimo de uma linguagem de programação quanto foi importante, na idade média, a disseminação do ensino da leitura e escrita que permitiu que pessoas do povo entendessem o que outros colocavam nos livros ou mesmo que escrevessem seus próprios livros, expressando suas idéias.

Infelizmente, esse projeto teve que parar este ano por problemas com a empresa financiadora. Mas, afirma Guido, esse projeto será continuado assim que possível.

O programa de lentes:

Formato de disponibilização

Sobre o formato, instalação e lugares de disponibilização, esclarecemos no Anexo I.

O programa em geral

Inicialmente, foi gasto em torno de três semanas no aprendizado de Python e adaptação à linguagem de programação e seus módulos gráficos e científicos como Numeric, SciPy e PyGame.

Depois desse curto período já foi possível começar a implementar o programa de lentes que foi entregue no dia 30 de abril sendo capaz de desenhar uma lente com diferentes curvaturas das superfícies (independentemente uma da outra), traçado de raios e diferentes posicionamentos da lente e o objeto.

Seria perfeitamente possível ter implementado diferentes indices de refração para a lente e diferentes meios para propagação da luz ao invés do ar ou vácuo, além de diferentes tamanhos (espessura e altura) da lente. Mas, foi esquecido de fazer essa pequena modificação antes de ser enviado ao grupo. No entanto, na próxima versão isso tudo deverá estar disponível junto com a possibilidade de colocar mais lentes.

O cálculo do traçado de raios é feito utilizando puramente a equação de refração:

$$n_i * \text{sen}(\Theta_i) = n_r * \text{sen}(\Theta_r) \qquad (1)$$

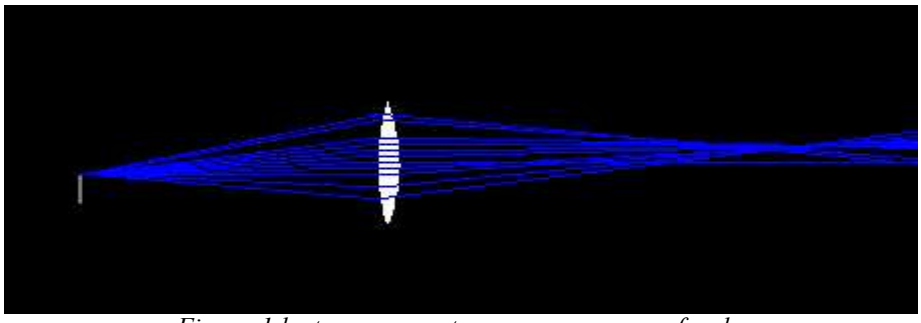

Figura 1 lente convergente com uma pequena focal

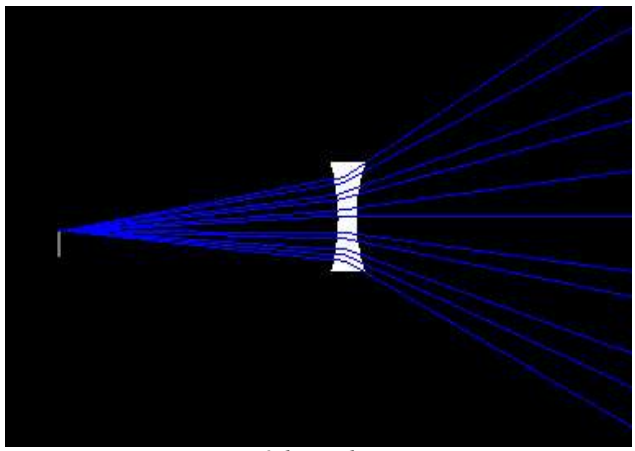

Figura 2 lente divergente

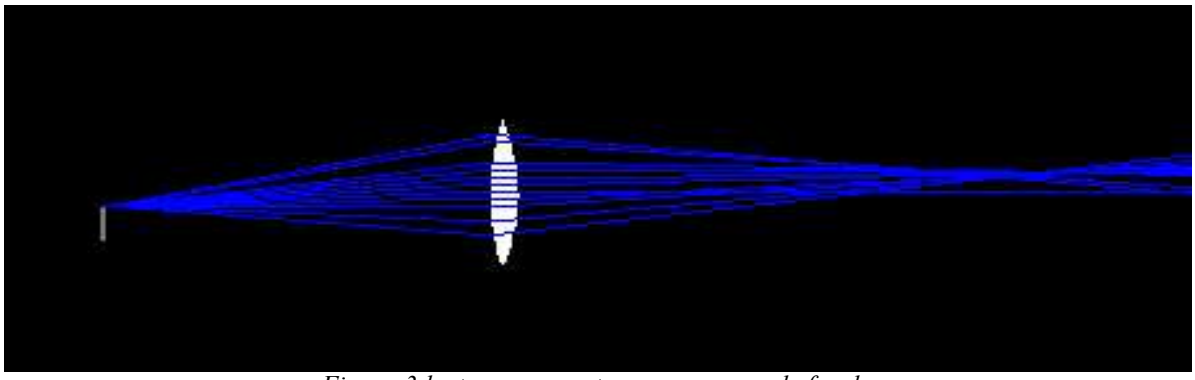

Figura 3 lente convergente com uma grande focal

O programa é capaz de realizar o traçado de raios para um número variado de lentes com diferentes índices de de refração, tamanho e largura, como é mostrado na figura 4. Nessa imagem, foram feitos comentários utilizando a cor branca, todo o resto é a janela do programa, como está atualmente.

As “posições marcadas”, como o próprio nome diz, são determinadas pelo usuário através das teclas “m” e “n” e, em conjunto com a tecla “ESC” que é utilizada para sair do programa, são as únicas teclas que o usuário deverá conhecer para utilizar totalmente o programa. O cálculo da distância entre esses dois pontos é feita sempre que existem esses dois pontos e é atualizada sempre que qualquer um dos pontos é modificado. Todas as outas funções estão apresentadas pelo menu na esquerda superior.

Por exemplo, quando não existem raios traçados, a opção “Apaga os raios” é substituída pela opção “Traçar os raios”. Clicando-se em cima de qualquer lente, pode-se obter as informações sobre ela e, clicando-se em “Apaga lente” apagaria essa lente. As informações disponíveis são posição, índice de refração, altura, largura, raio de curvatura da primeira superfície, raio de curvatura da segunda superfície e o número da lente.

Se pedir para adicionar uma lente ele, automáticamente, lhe informará um número que não está sendo utilizado para o número da lente, se você resolver alterar esse valor e colocar um valor já existente, ele alterará a lente com esse respectivo número.

Uma explicação mais detalhada estará disponível junto ao programa a partir do dia 16/06/2003.

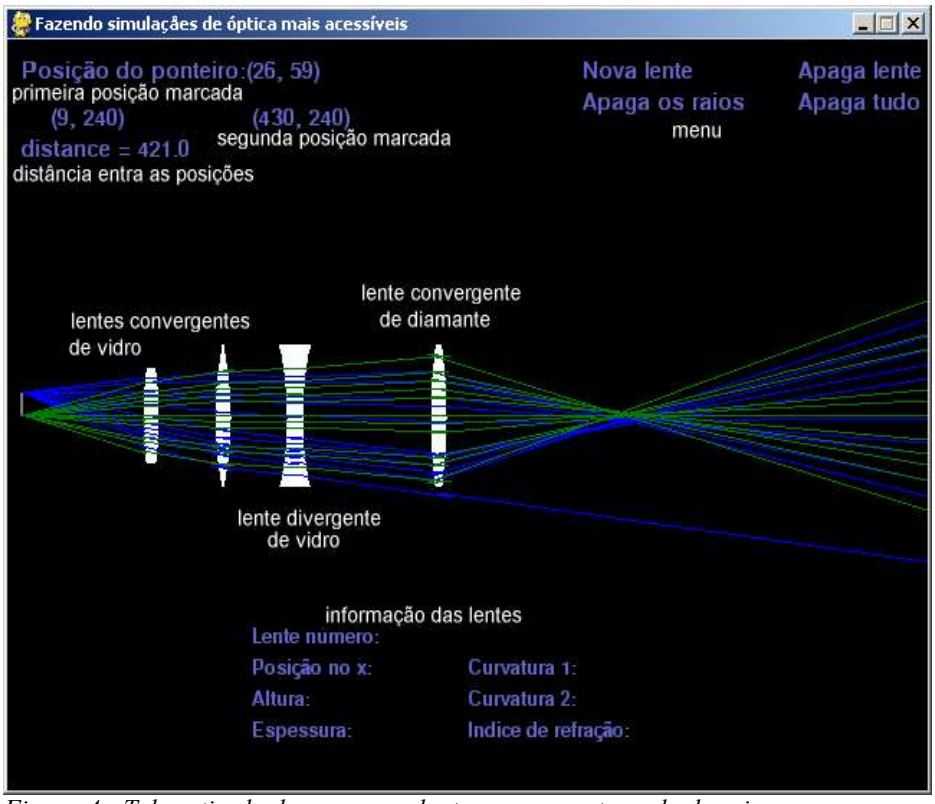

Figura 4 - Tela retirada do programa lentes.py, com o traçado de raios com diferentes tipos de lentes.

Uma abordagem mais teórica:

Quando temos um raio atingindo a superfície da lente (qualquer uma das duas superfícies), utilizamos a equação de refração (1). No entanto, essa equação é uma relação direta para o caso de um raio incidindo em uma superfície plana, e, vemos claramente, que a lente não é um plano (pode ter superfícies planas, mas não necessáriamente).

Por isso, é feito a seguinte análise:

1. Se o raio está chegando ou saindo da lente. Esse passo determinará qual valor de índice de refração será o do meio atual e qual será o do meio para o qual o raio está se dirigindo;
2. É determinado o ângulo de incidência e a direção normal à superfície, segundo a figura 5. O Plano pode ser conseguido, assumindo um valor muito alto para o raio de curvatura.

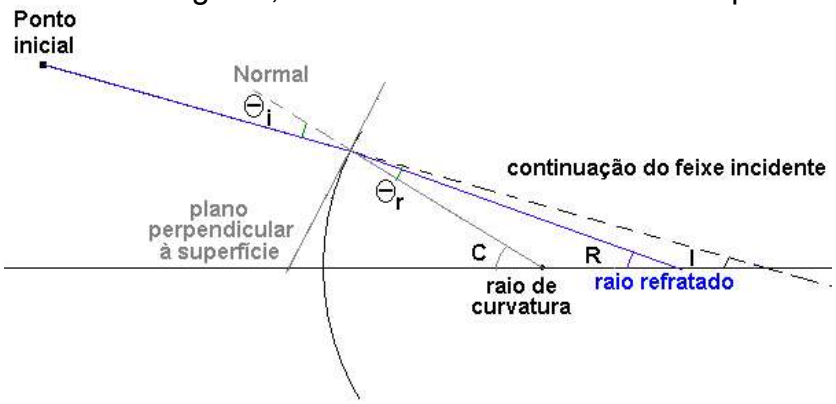

Figura 5 - esquema de raios em uma superfície esférica.

3. Determinamos, por último, a direção em que o raio vai prosseguir depois de atingir essa superfície. Se houver mais alguma superfície, o programa volta ao passo 1.

No entanto, a obtenção direta do valor de Θ_i e Θ_r é muito mais dificil, computacionalmente. Os únicos valores que obtemos diretamente são: o ângulo I, o raio de

curvatura (r) e o ponto onde a continuação do feixe incidente atinge o eixo traçado (d). Através de uma análise geométrica, podemos obter:

$$\text{Sen}(\Theta_i) = \text{Sen}(I)*(r-d)/d$$

Utilizando em (1), temos:

$$\text{Sen}(\Theta_r) = \text{Sen}(\Theta_i) *(n_r/n_i)$$

Com mais algumas análises, podemos obter R e C, para então obtermos o ponto onde o raio refratado toca o eixo traçado (que chamaremos ponto T).

$$R = I + \Theta_r - \Theta_i$$

E, finalmente,

$$T = ((\text{Sen}(\Theta_r)/\text{Sen}(R))*r) + r$$

Com isso, podemos obter o ângulo de saída de qualquer raio entrando ou saindo de uma lente esférica ou plana.

Considerações finais:

Além do que foi apresentado aqui e a modificação do programa para a inclusão de mais de uma lente na trajetória, temos o programa de difração por uma fenda e uma interface gráfica que reunirá os nossos dois programas de óptica e, provavelmente, programas de mecânica.

Este curso, além dos propósitos colocados no projeto, também está sendo uma boa forma de apresentar ao curso de física uma alternativa muito mais fácil, simples e poderosa do que a linguagem de programação Pascal, atualmente utilizada no ensino de introdução à computação à física. Vários outros lugares, dos quais posso citar com toda a certeza, a Universidade de Chicago e a PUCCamp, estão trocando ou já trocaram seus cursos de introdução à computação para Python com grande sucesso.

Idéias para projetos futuros:

Existe uma série de bibliotecas prontas que poderiam ser adicionadas à nossa interface com um pouco de trabalho. Entre estes estão uma biblioteca para resolução e visualização das soluções de problemas de eletromagnetismo, que pode ser obtido na página do pacote Numerical Python e um conjunto de ferramentas para criação dos mais diversos problemas de mecânica já com a interface e equações criadas. Tudo o que seria necessário seria estudar essas bibliotecas, sendo de possível implementação em cerca de um mês de trabalho.

Referências:

Bibliografia:

1. Python Tutorial – release 2.2.2, de Guido van Rossum, editado por Fred L. Drake, PythonLabs.
2. Principles of Optics, de Born & Wolf, sexta edição, Cambridge.

Internet:

1. http://www.gnu.org/
2. http://www.wxpython.org/
3. http://www.debian.org/
4. http://www.linux-france.org/prj/mek/
5. http://www.pythonemproject.com/
6. http://www.scipy.org/
7. http://www.opensource.org/
8. http://www.vex.net/parnassus/
9. http://www.pygame.org/
10. http://www.python.org/
11. http://www.vpython.org/
12. http://www.ibiblio.org/obp/thinkCSpy/

ANEXO

ANEXO I – Sobre a instalação e funcionamento do programa

O programa está sendo disponibilizado ao publico pela página do Teleduc, no portifólio de Marcelo de Freitas Rigon e na página:

http://sourceforge.net/projects/physics-edu

Não será distribuído a versão compilada (executável) pois teríamos que disponibilizar um para cada sistema operacional que gostaríamos que funcionasse, além de aumentar o tamanho do programa para um valor em torno de 20MB, para cada sistema operacional.

Liberando o código fonte, permite que qualquer pessoa utilize o programa mediante a obtenção de um programa e três pacotes de expansões para esse programa. Esses programas são gratuitos e não instalam nada indesejável no computador do usuário. Os componentes a serem instalados, nessa devida ordem, são:

1. Python 2.2 – http://www.python.org
2. PyGame para python 2.2–http://www.pygame.org
3. Numerical Python para python 2.2 –http://www.pfdubois.com/numpy/
4. SciPy para python 2.2 –http://www.scipy.org

Uma vez instalados esses programas, é só executar o programa Main.py da mesma forma que qualquer outro executável, se estiver utilizando um visualizador de arquivos como o windows explorer, ou, se estiver em modo linha de comando, é só digitar:

python Main.py

Um guia de como utilizar o programa estará incluso na versão 1.1 a ser disponibilizada na segunda-feira.