5
Estendendo as regiões da NCL 3.0

Em NCL, propriedades dos objetos de mídia definem em que regiões de uma tela esses objetos devem ser apresentados. Os valores iniciais dessas propriedades podem ser definidos nos elementos <property>, nos elementos <descriptorParam> ou nos elementos <region>. Na versão corrente, NCL 3.0, as regiões são definidas por meio de sua posição (*top* e *left* ou *botom*, *right*) e dimensão (*width* e *height*). Sendo que os atributos *top* e *height* têm preferência sobre *bottom*, bem como *left* e *width* têm preferência sobre *right*. Por meio desses atributos, é fácil perceber que, na versão atual, NCL só permite definir regiões bidimensionais e retangulares.

Como uma contribuição secundária desta dissertação e visando aperfeiçoar as regiões de NCL, este capítulo discute algumas alternativas de extensão para a linguagem que permitem aplicar transformações 2D e 3D nas regiões hoje existentes e, posteriormente, a definição de novas regiões como superfície de objetos tridimensionais. É importante ressaltar que a proposta deste capítulo não é tornar NCL uma linguagem para a descrição de cenas virtuais 3D, visto que ainda existem vários elementos, tais como iluminação, material, navegação, "áudio espacializado" – o volume e frequência do som cujo usuário percebe dependem da sua localização e movimento em relação à fonte – que não estão presentes nesta proposta. Assim como o restante deste trabalho, o objetivo ainda é manter NCL como uma linguagem de cola.

Cabe ressaltar também que não é foco deste trabalho escolher uma das alternativas de notação discutida neste capítulo para a linguagem NCL. Mesmo porque, para isso, é interessante que sejam realizadas pesquisas com os usuários da linguagem, o que não está no escopo desta dissertação. Por outro lado, o objetivo é definir funcionalidades que podem vir a ser incorporadas à NCL e discutir alternativas de notação para elas, comparando suas vantagens.

Este capítulo está dividido como segue. A Seção 5.1 discute algumas transformações 2D e 3D possíveis nas regiões NCL. Tais transformações

permitem distorcer as regiões retangulares de NCL, possibilitando efeitos que dão alguma sensação tridimensional. A Seção 5.3 discute como é possível incoporar os módulos de geometria de X3D e definir regiões em NCL como superfícies desses objetos.

5.1.Atributos de dimensão e posicionamento em NCL 3.0

Conforme comentado anteriormente, as regiões em NCL são especificadas por meio das propriedades dos objetos de mídia. Adicionalmente, com o objetivo de aumentar o reúso na linguagem, também é possível especificar a região inicial de exibição de um objeto de mídia nos descritores (elemento <descriptor>), por meio dos elementos <descriptorParam>, ou nos elementos <region>. O autor deve especificar as regiões em NCL por meio dos atributos *left*, *right*, *top*, *bottom*, *width* e *height*.

É importante ressaltar a diferença entre o que, neste texto, é chamado de região e o elemento <region>. Uma região é um conjunto de propriedades dos objetos de mídia que informam sua dimensão e posicionamento. O elemento <region> é apenas uma das formas de especificar essas regiões na linguagem NCL. Sua vantagem é que possibilita o reúso do mesmo valor dessas propriedades por mais de um objeto de mídia. Outra forma é diretamente por meio dos atributos <property> nos objetos de mídia, ou por meio do elemento <descriptorParam> nos descritores.

A Figura 42 demonstra como é possível definir as regiões de exibição dos objetos de mídia em NCL por meio de elemento <region> (a), descritores (b), e propriedades de objetos de mídia (c). Em (a) e (b) é possível reutilizar as mesmas propriedades para mais de um objeto de mídia. Adicionalmente, em (a) também é possível definir uma hierarquia de regiões que refletem semanticamente a disposição espacial dos objetos na tela.

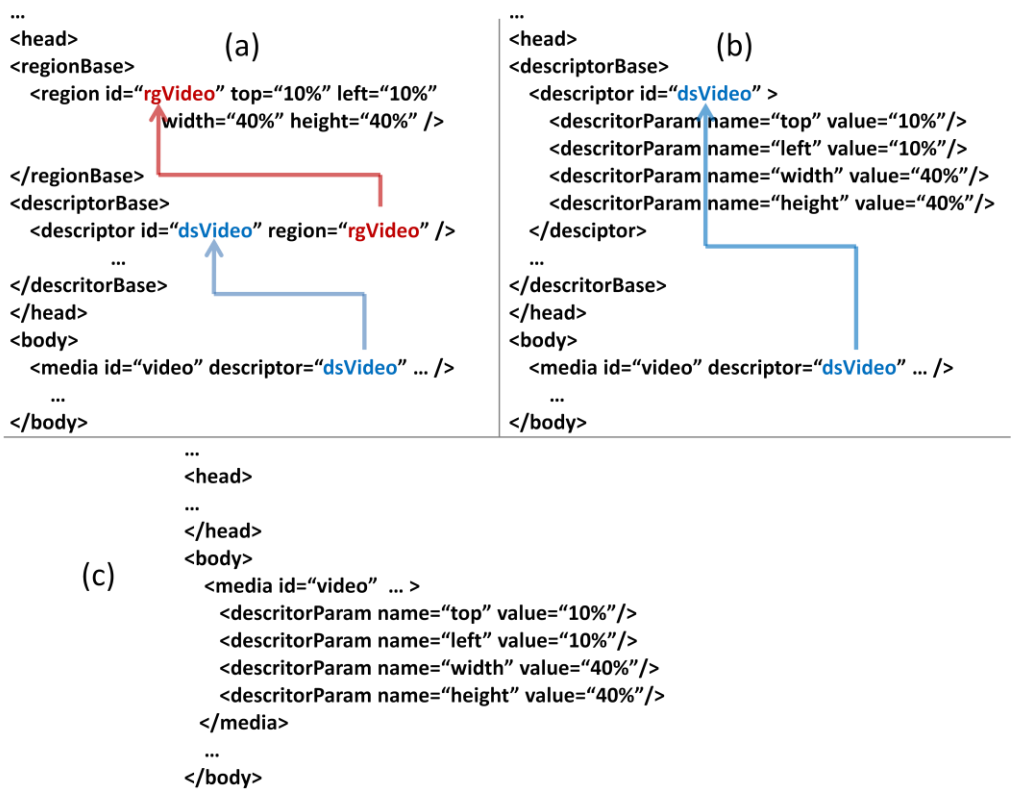

Figura 42 Definição de atributos de dimesão e posicionamento dos objetos de mídia por meio de: (a) regiões, (b) descritores e (c) propriedades de objetos de mídia.

Os elementos <region> também podem ser aninhados, sendo que os atributos de posicionamento e dimensão dos elementos <region> filhos são sempre relativos ao <region> pai. Por exemplo, se o atributo *top* de um elemento <region> é 10px (*pixels*), a região de exibição da mídia deve ser posicionada a 10px do topo do elemento <region> pai, e não a 10px do topo da tela. Caso o elemento <region> não possua um pai, a tela do dispositivo é considerada seu pai.

Adicionalmente, caso ocorra sobreposição de regiões, também é possível especificar qual objeto de mídia deve aparecer sobre outros por meio da propriedade *zIndex*. Objetos de mídia com valores de *zIndex* maiores devem aparecer sobre objetos de mídia com *zIndex* menores. A Figura 43 apresenta os atributos das regiões NCL (evidenciando a hierarquia de quando essas regiões são definidas por meio dos elementos <region>) e seus significados.

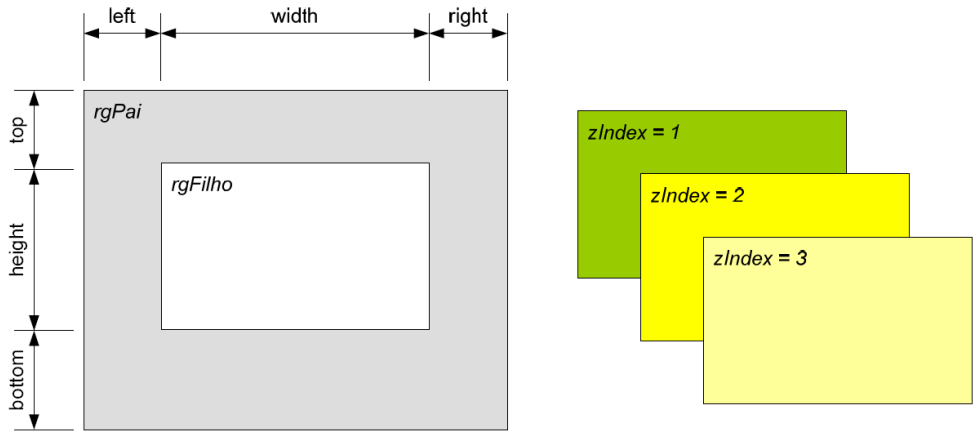

Figura 43 Atributos de posicionamento e dimensão de regiões na versão 3.0 de NCL. Fonte: (SOARES NETO, SOARES, *et al.*, 2010).

As seções a seguir discutem como é possível aperfeiçoar as regiões NCL aplicando transformações geométricas a elas (na Seção 5.2) e definindo-as como superfície de objetos tridimensionais (na Seção 5.2.2). Visando manter a mesma abordagem de NCL discutida acima, também é possível utilizar as propostas a seguir por meio de parâmetros em descritores ou como propriedades de objetos de mídia.

5.2.Transformações em regiões NCL

Conforme comentado, as regiões NCL 3.0 – o conjunto de propriedades que definem dimensão e posicionamento dos elementos de mídia – são estritamente retangulares e bidimensionais. Algumas transformações simples, entretanto, tais como rotação e cisalhamento podem prover aos autores de documentos multimídia um maior poder e efeitos bem mais sofisticados sobre os objetos de mídia que estão sendo apresentados. Esta seção propõe algumas transformações (tanto 2D como 3D) que aumentam o poder da linguagem e discute como essas transformações podem ser definidas como propriedades de objetos de mídia, descritores ou regiões.

Para permitir a definição das transformações propostas a seguir são sugeridas duas notações alternativas e analisadas as vantagens e desvantagens de cada uma delas. São elas:

(i) *Uma nova propriedade* (*transform*), a ser adicionada como atributo ao elemento <region>, como parâmetro aos descritores (especificado

por meio do elemento <descriptorParam>) e como propriedade (elemento <property>) aos objetos de mídias.

(ii) A definição de *propriedades específicas* para cada uma das transformações permitidas nas regiões. No elemento <region> um novo atributo pode ser adicionado para cada uma das transformações. Da mesma forma, um novo parâmetro pode ser adicionado aos descritores e uma nova propriedade (elemento <property>) às propriedades de mídias.

Na notação (i) a propriedade *transform* deve possuir uma lista de transformações a serem aplicadas, em sequência, à região de exibição do objeto de mídia. Essa abordagem é similar à de CSS3 (W3C, 2001). Entretanto, as transformações disponíveis e os parâmetros das transformações sugeridos nesta dissertação diferem da proposta de CSS3, conforme é discutido no decorrer desta seção. Como um exemplo, algumas das transformações de CSS3 não são sugeridas quando já é possível realizá-las de outra forma em NCL. Isso objetiva manter a simplicidade da linguagem NCL.

A Figura 44 apresenta um exemplo de como definir uma transformação por meio do atributo de uma região (linha 2), de um parâmetro de descritor (linhas 6 a 8) e de uma propriedade de um objeto de mídia (linhas 12 a 14). O valor atributo *transform* deve ser uma lista de tranformações que devem ser aplicada em *sequência* àquela região.

```
1: <!-- Especificação de Transformações em Regiões -->
2: <region id="rg01" width="100%" height="150%"
              transform="rotate(90deg) skew(10deg)"/>
3:
4: ...
5: <!-- Especificação de Transformações como parametro de
   Descritor -->
6: <descriptor id="ds01">
7:   <descriptorParam name="transform"
                    value="rotate(90deg) skew(10deg)"/>
8: </descriptor>
9:
10:...
11:<!-- Especificação de Transformações como propriedade de
   objeto de mídia -->
12:<media id="md01" ... >
13:  <property name="transform"
              value="rotate(90deg) skew(10deg)"/>
14:</media>
```

Figura 44 Definição de transformações em regiões, descritores e propriedades de objetos de mídia em NCL através do parâmetro *transform* – notação (i).

Na notação (ii), exemplificada na Figura 45, por outro lado, as transformações devem ser definidas como parâmetros individuais do elemento <region> (linha 2). Da mesma forma, ela apresenta como é possível definir parâmetro em descritores (linhas 6 a 8) e propriedades de mídia (linhas14 a 17). Essa notação parece ser mais simples para usuários da linguagem que não são programadores, visto que possui um caráter mais próximo de uma linguagem declarativa do que (i).

```
1: <!-- Especificação de Transformações em Regiões -->
2: <region id="rg01" width="100%" height="150%"
              rotation="90deg" skew="10deg" />
3:
4: ...
5: <!-- Especificação de Transformações como parametro de
   Descritor -->
6: <descriptor id="ds01">
7:   <descriptorParam name="rotation"
                     value="90deg" />
8: <descriptorParam name="skew"
                     value="10deg" />
9:
10:</descriptor>
11:
12:...
13:<!-- Especificação de Transformações como propriedade de
   objeto de mídia -->
14:<media id="md01" ... >
15:  <property name="rotation"
              value="90deg"/>
16:  <property name="skew"
              value="10deg"/>
17:</media>
```

Figura 45 Definição de transformações em regiões, descritores e propriedades de objetos de mídia em NCL, por meio de atributos específicos para cada transformação – notação (ii).

Em (i) é fácil perceber que o autor deve informar *passo a passo* quais as tranformações que devem ser aplicadas, e *em qual ordem*. Enquanto em (ii) essa ordem já deve ser pré-definida pela linguagem. Sendo assim, (i) é mais expressiva do que (ii), já que o autor pode definir a ordem que lhe convier. Por exemplo, em (i) é possível especificar mais de uma transformação de rotação, enquanto em (ii) isso não é permitido.

Outro fato que também merece ser mencionado é que (i) exige um *parser* específico para o atributo *tranform*, já que esse atributo não é atômico. Em (ii) isso é amenizado, pois, no máximo, o atributo XML introduzido na linguagem possui uma lista de valores, seguindo o mesmo padrão de algumas propriedades de NCL (tais como *bounds*, *location*, entre outros.). Nesse útimo caso, o próprio

parser XML pode ser utilizado para descobrir quais as transformações a serem aplicadas.

5.2.1.Transformações 2D em regiões NCL

Para permitir a definição de transformações geométricas nas regiões NCL, inicialmente, se faz necessário que exista um sistema de coordenadas. Visando não interferir em como a versão atual de NCL define regiões, é interessante que elas continuem sendo definidas por meio de sua altura, largura, posição do topo etc. Ainda visando manter a compatibilidade com a versão atual, também é interessante que o atributo *zIndex* mantenha sua semântica mesmo em transformações 3D, ou seja, regiões com *zIndex* maiores devem aparecer sobre regiões com *zIndex* menores.

Uma alternativa viável para manter a compatibilidade com a versão atual de NCL é definir um sistema de coordenadas local para cada região e aplicar as transformações nesse sistema de coordenadas. Por *default*, para as transformações 2D é interessante que a origem do sistema de coordenadas seja o centro da região NCL (50%, 50%). A Figura 46 apresenta os eixos x e y desse sistema de coordenadas.

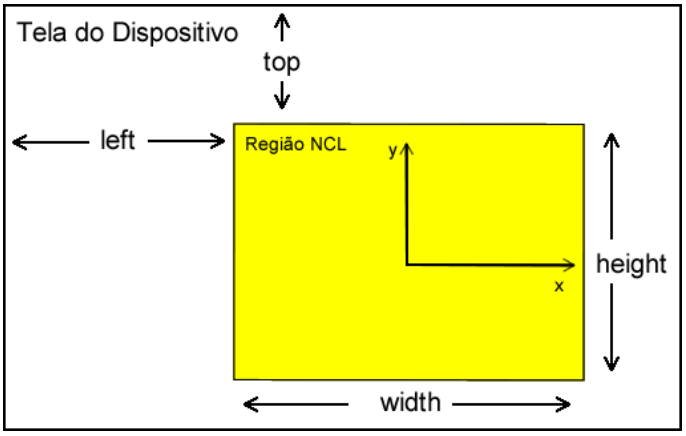

Figura 46 Sistema de coordenadas 2D em regiões NCL.

A Tabela 3 apresenta quais as transformações sugeridas por esta dissertação e que podem ser adicionadas segundo a proposta (i). Enquanto a Tabela 4 apresenta os novos atributos para o elemento <region> segundo a proposta (ii), discutida anteriormente.

Tabela 3 Novas transformações 2D para rotação, cisalhamento e espelhamento, seguindo a proposta (i).

Transformações 2D	Descrição	Valores Possíveis
rotate(angle)	Rotaciona uma região em um ângulo (*angle*) no sentido anti-horário.	*angle* pode ser especificado em graus (deg), radianos (rad) ou porcentagem (%). Caso a unidade de medida não seja informada, deve-se assumir que os valores são em graus.
skew(angleX, [angleY])	Aplica transformação de cisalhamento com um ângulo de *angleX* no eixo X e *angleY* no eixo Y. *angleY* é opcional.	*angleX* e *angleY* devem ser especificados em graus (deg), radianos (rad) ou porcentagem (%). Caso a unidade de medida não seja informada, deve-se assumir que os valores são em graus.
flipHorizontal()	Inverte (espelha) o conteúdo que será apresentado nessa região no eixo horizontal (x).	
flipVertical()	Inverte (espelha) o conteúdo da que será apresentado nessa região no eixo vertical (y).	

Tabela 4 Novos atributos para rotação, cisalhamento e espelhamento, seguindo a proposta (ii).

Novos atributos da região para tranformação 2D	Descrição	Valores Possíveis
rotation	Rotaciona uma região no ângulo definido pelo valor desse atributo, no sentido anti-horário.	O valor desse atributo deve ser um ângulo, especificado em graus (deg), radianos (rad) ou porcentagem (%). Caso a unidade de medida não seja informada, deve-se assumir que os valores são em graus.
skew	Aplica transformação de cisalhamento no eixo X e no eixo Y.	O valor desse atributo deve ser dois números reais com alguma unidade separados por virgulo: *angleX* e *angleY*. *angleX* e *angleY* devem ser especificados em graus (deg), radianos (rad) ou porcentagem (%). Caso a unidade de medida não seja informada, deve-se assumir que os valores são em graus.
flipHorizontal	Caso o valor seja *true*, inverte (espelha) o conteúdo da mídia que será apresentado no eixo	*true* ou *false*.

	horizontal (x).	
flipVertical	Caso o valor seja *true*, inverte (espelha) o conteúdo da mídia que será apresentado no eixo vertical (y).	*true* ou *false*

CSS3 não possui as tranformações de espelhamento (*flipVertical* e *flipHorizontal*) apresentadas acima. Por outro lado, possui transformações de translação e escala. Em 2D, NCL permite que translação e escala sejam realizadas alterando-se os valores de *top*, *left*, *bottom* e *right* (para translação), ou *width* e *height* (escala). Por isso, elas podem ser desconsideradas. As subseções a seguir discutem detalhadamente cada uma das transformações propostas.

5.2.1.1.Rotação 2D

Para rotacionar um objeto em duas dimensões (como são as regiões de NCL hoje) é necessário e suficiente que se defina um *ponto* e *um ângulo de rotação*. Por simplicidade, pode-se considerar o ponto de rotação como sendo a origem do sistema de coordenadas, conforme definido anteriormente.

A Figura 47 apresenta um exemplo de rotação 2D em uma região NCL segundo as notações (i) e (ii), discutidas no início desta seção. Conforme apresentado na Tabela 3 e na Tabela 4, o ângulo de rotação pode ser definido em graus, radianos ou porcentagem (100% equivale a uma rotação de 360deg).

Seguindo a notação (i), também é possível especificar essa transformação por meio do elemento <descriptorParam>, no descritor, ou por meio do elemento <property>, no objeto de mídia. Nesses dois últimos casos, o atributo *name*, do elemento <descriptorParam> ou <property>, deve ser igual a *transform* e o atributo *value* deve ser igual ao valor da transformação especificado no elemento <region>, seguindo o mesmo padrão exemplificado na Figura 44.

(i)
```
<region id="rgPai" width="40%" height="40%"
                   transform="rotate(45deg)">
...
</region>
```

(ii)
```
<region id="rgPai" width="40%" height="40%" rotation="45deg">
 ...
</region>
```

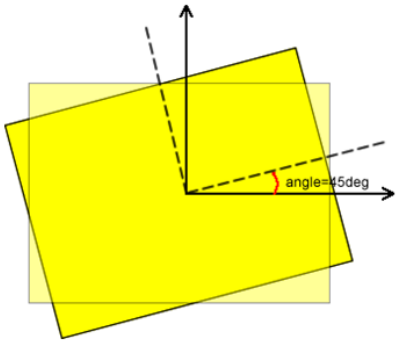

Figura 47 Rotação de região NCL especificada no elemento <region> segundo a notação (i) e (ii).

5.2.1.2.Cisalhamento 2D

O cisalhamento (em inglês *shear* ou *skew*) é uma transformação que distorce objetos. Em 2D, o objetivo é deslocar as coordenadas em x ou y, linearmente, segundo um fator definido pelo autor. Em NCL, assim como em CSS3, é sugerido que esse fator de deslocamento seja definido por meio de um ângulo entre o eixo de origem e o "novo eixo". Como discutido anteriormente, a origem das tranformações geométricas aqui discutidas é o centro da região. A Figura 48 apresenta como é possível definir uma transformação de cisalhamento em NCL, segundo as notações (i) e (ii).

Conforme apresentado nas Tabelas 3 e 4, essa tranformação pode ser adicionada à NCL por meio da transformação *skew(angle)*, na propriedade *transform*, seguindo a notação (i), ou por meio da propriedade *skew*, seguindo a notação (ii). O ângulo de cisalhamento pode ser definido em graus, radianos ou porcentagem (100% equivale a um cisalhamento 90deg).

A Figura 48 apresenta um exemplo de definição de uma transformação de cisalhamento em uma região de exibição de um objeto de mídia NCL por meio de parâmetros nos descritores. Da mesma forma que em todas as transformações aqui apresentadas, também é possível especificá-la no elemento <region> ou como propriedade (elemento <property>) no objeto de mídia.

(i)
```
<descriptor id="ds01">
  <descritorParam name="transform" value="skew(10deg)">
</descriptor>
```

(ii)
```
<descriptor id="ds01">
  <descritorParam name="skew" value="10deg">
</descriptor>
```

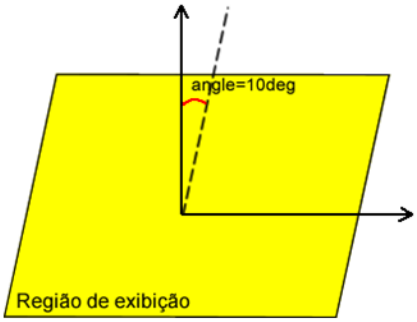

Figura 48 Exemplo de definição de transformação de cisalhamento em NCL, por meio de parâmetros nos descritores, segundo as notações (i) e (ii).

5.2.1.3.Espelhamento 2D

A transformação de espelhamento permite que o conteúdo que será apresentado em uma região de exibição apareça invertido de alguma forma, segundo o que o autor desejar. O espelhamento pode ser realizado tanto no eixo x, como no eixo y local da região. Como é discutido na Subseção 5.2.2.1, o mesmo efeito de espelhamento em 2D pode ser obtido com a rotação de 180° no eixo y, em uma transformação 3D.

A Figura 49 apresenta o resultado da aplicação apenas do espelhamento vertical (*flipVertical*), do espelhamento horizontal (*flipHorizontal*) e, dos dois espelhamentos em conjuntos (*flipHorizontal+flipVertical*) em uma imagem.

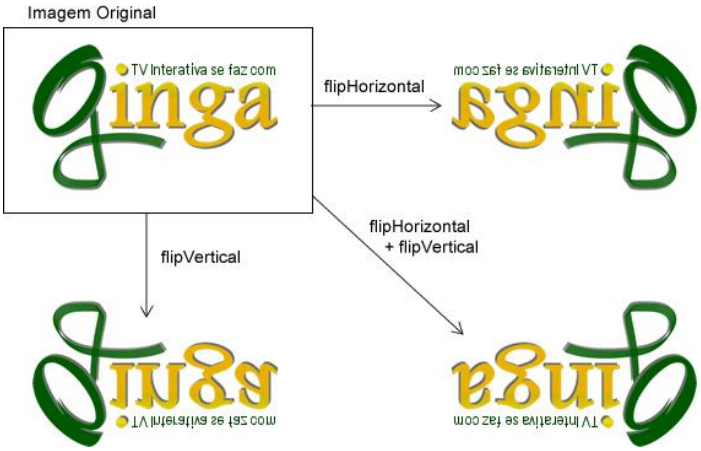

Figura 49 Demonstração da transformação de *flipHorizontal* e *flipVertical* em imagem.

A Figura 50 exemplifica como o autor do documento pode definir o espelhamento vertical e horizontal em uma região de exibição de um objeto de mídia em NCL. Este exemplo apresenta como é possível definir a transformação de espelhamento por meio do elemento <property> diretamente no objeto de mídia. Assim como nas transformações anteriores é possível defini-la também no elemento <region>, assim como no elemento <descriptorParam>.

Na notação (i) ambos são definidos como funções que podem ser adicionadas à propriedade *transform* e não recebem parâmetro algum. Enquanto na notação (ii) são propriedades *booleanas* (verdadeiro ou falso) dos objetos de mídia. Em (ii), por omissão (*default*), os valores das propriedades *flipVertical* e *flipHorizontal* são *false*.

(i)
```
<media id="myMedia" ... >
  <property name="transform" value="flipVertical()
                                    flipHorizontal()">
</media>
```

(ii)
```
<media id="myMedia" ... >
  <property name="flipVertical" value="true"/>
  <property name="flipHorizontal" value="true"/>
</media>
```

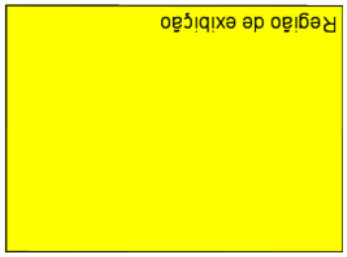

Figura 50 Exemplo de definição de espelhamento vertical e horizontal segundo as notações (i) e (ii).

5.2.2.Transformações 3D em regiões NCL

Embora as transformações discutidas na subseção anterior permitam rotacionar, distorcer e espelhar as regiões 2D de NCL, elas ainda não são sucientes para passar alguma noção de profundidade. Tal noção de profundidade pode ser suportada por meio de transformações 3D. A definição dessas tranformações 3D pode ser vista como um passo intermediário entre a definição de regiões NCL 2D, como são hoje, e de regiões como superfície de objetos 3D, que é o objetivo deste capítulo.

Cabe ressaltar que mesmo de posse das transformações 3D discutidas nesta subseção, as regiões NCL continuam sendo bidimensionais. Elas sempre serão um plano. Mas que podem ser distorcidos ou rotacionados permitindo algum efeito de profundidade.

Conforme discutido anteriormente, com o objetivo de manter a compatibilidade com as regiões atuais, todas as nossas transformações devem ser realizadas em um sistema de coordenadas local, para cada região. A Figura 51 apresenta o sistema de coordenadas cartesianas 3D local para uma região NCL que, por *default*, encontra-se no centro da região. A origem do sistema de coordenadas 3D de uma região é no ponto central dessa região e segue a regra da mão direita, assim como demonstrado na Figura 51.

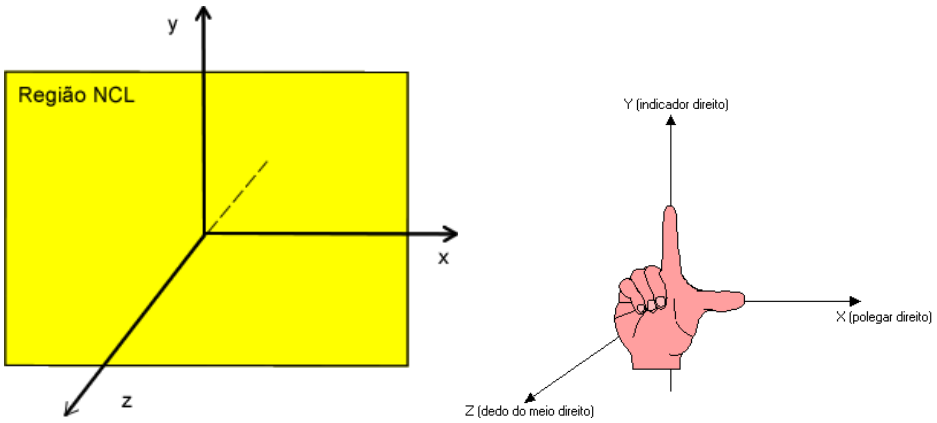

Figura 51 Sistema de coordenadas cartesianas local de uma região (segundo regra da mão direita) tomado como base para transformações tridimensionais.

Assim como para as transformações 2D, nesta subseção também são discutidas as duas notações propostas no início desta seção. A Tabela 5 apresenta as sugestões de transformações 3D segundo a notação (i) e a Tabela 6 segundo a notação (ii). Essas transformações possuem, propositalmente, o mesmo nome das transformações 2D, sendo que é possível distiguí-las apenas pelo número de argumentos, na notação (i) e o formato do atributo na notação (ii).

Tabela 5 Novas transformações 3D, seguindo a proposta de notação (i).

Transformações 3D	Descrição	Valores Possíveis
translate(x, y, z)	Translada a região de exibição para a posição x, y, z do sistema de coordenadas local da região NCL.	*x*, *y* e *z* devem ser valores reais.
rotate(x, y, z, angle)	Rotaciona uma região ao redor de um vetor (x, y, z) em um ângulo de: *angle*, no sentido anti-horário.	*x*, *y* e *z* devem ser valores reais. *angle* deve ser especificado em graus (deg), radianos (rad) ou porcentagem (%).

		Caso a unidade de medida não seja informada, deve-se assumir que os valores são em graus.
skew(angleX, [angleY], [angleZ])	Aplica transformação de cisalhamento com um ângulo de *angleX* no eixo X, *angleY* no eixo Y e *angleZ* no eixo Z. *angleY* e *angleZ* são opcionais.	*angleX*, *angleY* e *angleZ* deve ser especificados em graus (deg), radianos (rad) ou em porcentagem (%). Caso a unidade de medida não seja informada, deve-se assumir que os valores são em graus.
scale(x,y,z)	Redimensiona a região de apresentação da mídia segundo um fato de x no eixo X, y no eixo Y e z no eixo Z.	x, y e z devem ser valores reais.

Tabela 6 Novos atributos para transformações 3D, seguindo a proposta de notação (ii).

Novos atributos da região para tranformação 2D	**Descrição**	**Valores Possíveis**
translation	Translada a região NCL para a posição especifica pelo valor desse atributo	O valor desse atributo deve ser uma lista com três números reais separados por vírgula: "x, y, z".
rotation	Rotaciona uma região ao redor de um vetor, no sentido anti-horário. O vetor e o ângulo de rotação são informados como valore desse atributo.	O valor desse atributo deve ser uma lista com quatro valores, separados por vírgula: "*x, y, z, angle*". *x, y* e *z* devem ser valores reais. *angle* deve ser especificado em graus (deg), radianos (rad) ou porcentagem (%). Caso a unidade de medida não seja informada, deve-se assumir que os valores são em graus.
skew	Aplica transformação de cisalhamento nos eixo X, Y, Z, segundo informado pelo seu valor.	O valor desse atributo deve ser três números reais com alguma unidade separados por vírgula: "*angleX*, *angleY*, *angleZ*". *angleX*, *angleY* e *angleZ* devem ser especificados em graus (deg), radianos (rad) ou porcentagem (%). Caso a unidade de medida não seja informada, deve-se assumir que os valores são em graus.
scale	Redimensiona a região de apresentação da mídia segundo o	O valor desse atributo deve possuir três números

	que é informado pelo seu valor.	reais separados por vírgula: “x, y, z”

5.2.2.1.Translação 3D

A translação 3D permite modificar a posição de exibição de um objeto de mídia no sistema de coordenadas local daquela região. Por exemplo, é possível distanciar o objeto, o que resulta em ele aparecer menor na apresentação NCL. A Figura 52 apresenta um exemplo de translação na região NCL, reposicionando em (0,0,-10) no seu sistema de coordenadas local.

```
(i)  <region id="rgPai" width="40%" height="40%"
                    transform="translate(0,0,-10)">
     </region>

(ii) <region id="rgPai" width="40%" height="40%"
                    translation="0,0,-10">
     </region>
```

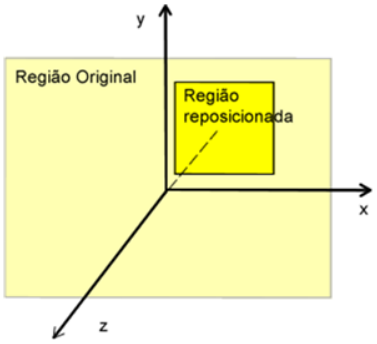

Figura 52 Exemplo de translação de uma região de exibição de um objeto de mídia especificada no elemento <region> NCL.

Seguindo a mesma abordagem das transformações 2D, também é possível especificar essa tranformação por meio de parâmetros de descritores ou elementos <property> nos objetos de mídia.

5.2.2.2.Rotação 3D

Existem duas formas comuns de descrever qualquer rotação em três dimensões:

a) três ângulos (*angleX*, *angleY* e *angleZ*), representando três rotações, uma ao redor de cada eixo do sistema de coordenadas. Esses ângulos são denominados ângulos de Euler.
b) Um vetor e um ângulo de rotação ao redor desse vetor.

Usando os ângulos de Euler, uma sequência diferente de aplicação das rotações resulta em resultados diferentes. Por isso, faz-se necessário definir a ordem em que as rotações de cada eixo devem ser aplicadas. Uma ordem comum é: primeiro a rotação no eixo X, depois no eixo Y e, por fim, no eixo Z. Outro problema com os ângulos de Euler é que uma interpolação (o que é bastante útil para fazer animação, também em NCL) entre seus valores, não necessariamente gera posições intermediária que estão entre a origem e o destino.

Por esses motivos, este trabalho sugere a utilização da segunda forma de descrição de rotação, por meio de um vetor e um ângulo de rotação ao redor desse vetor. Apenas relembrando, todas as transformações propostas têm origem no centro da região. A Figura 53 apresenta os parâmetros necessários para definir essa transformação.

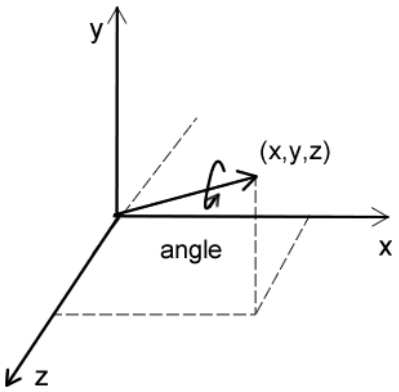

Figura 53 Parâmetros para a definição de uma rotação em uma região NCL.

A Figura 54 apresenta o código NCL exemplificando como definir uma transformação de rotação em uma região de exibição de um objeto de mídia, por meio do elemento <descriptorParam>. Esse exemplo demonstra uma rotação ao redor do eixo Y, isto é, vetor de rotação (0,1,0) e ângulo de 90 graus. São apresentadas as notações (i) e (ii), juntamente com o efeito visual da rotação da região.

(i)
```
<descriptor id="ds01">
<descritorParam name="transform" value="rotate(0,0,1,90deg)"/>
</descriptor>
```

(ii)
```
<descriptor id="ds01">
  <descritorParam name="rotation" value="0,0,1,90deg"/>
</descriptor>
```

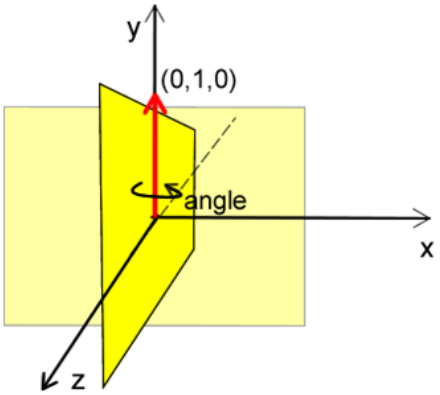

Figura 54 Região NCL com rotação de 90 graus ao redor do eixo Y.

5.2.2.3.Cisalhamento 3D

De forma semelhante ao cisalhamento 2D, o cisalhamento 3D também permite a definição de um fator de distorção linear ao longo dos eixos. A única diferença é que agora também é possível distorcer a região ao longo do eixo Z.

5.3.Definição de regiões NCL como superfície de objetos 3D

Mesmo com as tranformações 2D e 3D discutidas até o momento, ainda é fato que as regiões NCL são bidimensionais. Elas ainda são um plano, que, no máximo, podem ser distorcidos ou rotacionados. Uma abordagem mais poderosa é possibilitar a definição de regiões NCL como superfície de objetos 3D. Esta seção discute como definir alguns objetos geométricos básicos como regiões NCL e renderizar os objetos de mídia NCL nessas regiões.

Para definir regiões NCL como superfície de objetos geométricos tridimensionais, faz-se necessário que a linguagem ofereça algum mecanismo nas regiões que permita a descrição desses objetos geométricos. Um atributo *geometry*, por exemplo, no elemento <region>, possívelmente, é suficiente para isso, conforme demonstrado na Figura 55. As dimensões desse objeto podem ter um valor *default* sendo que o autor pode livremente utilizar as tranformações

discutidas na seção anterior para redimensionar ou reposicionar o objeto no sistema de coordenadas local daquela região.

```
1: <!-- Especificação de Região como superfície de uma caixa 3D
   no elemento region -->
2: <region id="rg01" width="100%" height="150%"
            geometry="box"
            transform="scale(10,20,30) rotation(0,1,0,45deg)"/>
3:
4: ...
5: <!-- Especificação de Região como superfície de uma caixa 3D
   por meio de parametros de Descritores -->
6: <descriptor id="ds01">
7:   <descriptorParam name="geometry"
                     value="box"/>
8:   <descriptorParam name="transform"
                value="scale(10,20,30) rotation(0,1,0,45deg)"/>
9: </descriptor>
10:
11:...
12:<!-- Especificação de Região como superfície de uma caixa 3D
   por meio de propriedades de mídia -->
13:<media id="md01" ... >
14:  <property name="geometry"
              value="box"/>
15:  <descriptorParam name="transform"
               value="scale(10,20,30) rotation(0,1,0,45deg)"/>
16:
17:</media>
```

Figura 55 Definição de regiões NCL como superfície de objetos 3D por meio do atributo *geometry*.

Outra possibilidade é reutilizar os módulos que definem elementos geométricos em X3D e embuti-los como filhos do elemento <region>, como demonstrado na Figura 56. Nesse caso, as definições de parâmetros de descritores e propriedades de objetos de mídia podem permanecer iguais ao que foi apresentado na Figura 55.

A única ressalva a se fazer é que quando X3D informa uma lista de valores, ela o faz separando-os por espaço (por exemplo, “0 0 0”). Para manter o mesmo padrão do resto da linguagem NCL, tais listas podem ter o valor separado por vírgula, conforme é possível observar para o atributo *size* do elemento *Box* abaixo. Nesse caso, também é possível especificar os valores de dimensão diretamente no elemento que define a geometria.

```
1: <!-- Especificação de Região como superfície de uma caixa 3D
   no elemento region (reutilizando módulo de geometria de X3D)
2: -->
3: <region id="rg01" width="100%" height="150%">
4:   <Box size="1.0, 3.0, 1.0"/>
5: </region>
```

Figura 56 Especificação de uma região NCL como a superfície de objetos geométricos reutilizando os módulos de geometria de X3D.

Mais uma vez, visando manter a compatibilidade com a versão atual, deve-se utilizar o sistema de coordenadas local de cada região para posicionar esses objetos geométricos. O atributo *zIndex*, também ainda deve manter sua semântica, objetos de mídia com *zIndex* maiores devem aparecer sobre objetos de mídia com *zIndex* menores, não importando a sua posição no eixo Z no sistema de coordenadas local. Caso duas regiões tenham o mesmo *zIndex*, o posicionamento no eixo Z pode indicar qual objeto de mídia deve aparecer sobreposto.

Por *default*, o objeto de mídia deve aparecer replicado em cada uma das faces do objeto geométrico. Assim, um vídeo sendo apresentado em cubo, que possui 6 faces, por exemplo, deve aparecer em cada uma das faces desse objeto. Como um dos trabalhos futuros desta dissertação, é importante que mecanismos que permitam ao autor definir como esses objetos de mídia serão mapeados nessas novas regiões estejam disponíveis na linguagem.

Outro ponto que também merece um estudo mais detalhado é a definição de um sistema de coordenadas 3D para as regiões NCL, tornando realmente NCL uma linguagem 3D. Isso possivelmente acabaria com a necessidade dos sistemas de coordenadas locais, aqui discutidos. Por outro lado, traz vários impactos na linguagem que não se restringem à definição das regiões.

Em primeiro lugar, a linguagem deverá posuir mecanismos para definições de câmera e, possivelmente, um avatar (incluindo suas dimensões) do usuário. Permitindo, inclusive sua movimentação, navegação na cena. Fora isso, também é necessário um estudo mais detalhado sobre como ocorre a interação entre os exibidores de mídia neste novo ambiente.

Atualmente, cada exibidor está restrito a sua região de exibição – isso ainda é valido, mesmo com tudo que foi discutido neste capítulo. Seguindo a mesma lógica, em um ambiente 3D, talvez devêssemos especificar uma região como um volume envolvente, na qual o exibidor só pode renderizar dentro desse volume. Contudo, isso traz vários problemas que devem ser estudados em maiores detalhes. Por exemplo: O que acontece quando ocorre a interseção entre regiões?

Talvez, o próprio conceito de região que existe hoje em NCL, não seja prático em um ambiente 3D. O mais interessante provavelmente é apenas definir o posicionamento dos objetos (cujo conteúdo não deve ser definido em NCL). Suas dimensões também devem ser inatas do próprio objeto de mídia (o que inclusive

já são com as resoluções de vídeo e imagens). NCL pode, no máximo, permitir a modificação das dimensões desses objetos de mídia.

Para manter a compatibilidade com a versão atual, as regiões de NCL 3.0 hoje, podem ser tratadas como um plano de exibição que acompanha a movimentação desse usuário. Contudo, isso também merece um estudo mais detalhado.