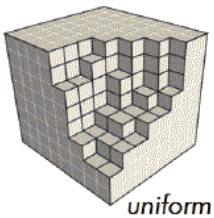
10. 3D-Strömungsvisualisierung

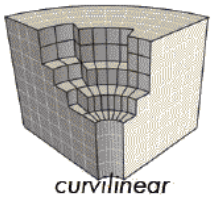
3D Vektorfelder

- *Definition: 3D Vektorfeld*

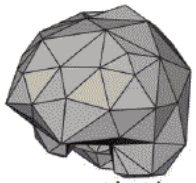
$$\vec{v}(\vec{x}) = \begin{pmatrix} v_x(x,y,z) \\ v_y(x,y,z) \\ v_z(x,y,z) \end{pmatrix} \qquad \vec{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$



uniform



curvilinear



unstructured

Jacobi-Matrix in 3D

- Wie stark *ändert* sich das Vektorfeld an der Stelle $\vec{x}$?

  Totales Differential: Jacobi-Matrix *(analog zu 2D)*

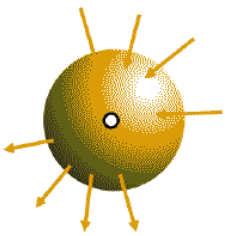
$$\mathbf{J}_{\vec{v}}(\vec{x}) = \begin{pmatrix} \frac{\partial v_x}{\partial x} & \frac{\partial v_x}{\partial y} & \frac{\partial v_x}{\partial z} \\ \frac{\partial v_y}{\partial x} & \frac{\partial v_y}{\partial y} & \frac{\partial v_y}{\partial z} \\ \frac{\partial v_z}{\partial x} & \frac{\partial v_z}{\partial y} & \frac{\partial v_z}{\partial z} \end{pmatrix}$$

Richtungsableitung: Wie stark ändert sich das Vektorfeld in Richtung $\vec{h}$?

$$\frac{\partial \vec{v}(\vec{x})}{\partial \vec{h}} = \mathbf{J}_{\vec{v}}(\vec{x})\,\vec{h}$$

Divergenz in 3D

- Analog zu 2D: Summe der Elemente der Hauptdiagonale der Jacobi-Matrix



$$\operatorname{div} \vec{v}(\vec{x}) = \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z}$$

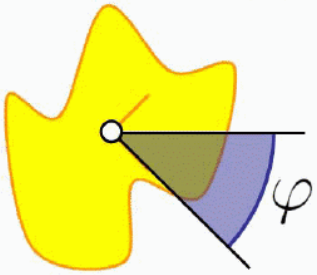
Interpretation:

- Betrachte ein infinitesimal kleines Volumen um den Punkt
- Wieviel fließt hinein, wieviel hinaus?

$\operatorname{div} \vec{v}(\vec{x}) < 0$	$\operatorname{div} \vec{v}(\vec{x}) > 0$
Senke, „Zusammenströmen“	Quelle, „Auseinanderströmen“

Rotation in 2D

- *Aus der Schulphysik: Rotation starrer Körper*



Ein Körper dreht sich in der Zeit t um den Winkel φ.

Der Körper dreht sich mit der Winkelgeschwindigkeit ω:

Bei konstanter Winkelgeschwindigkeit: $\omega = \dfrac{\varphi}{t}$

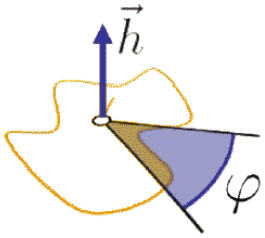
Bei zeitabhängiger Winkelgeschwindigkeit: $\omega(t) = \dfrac{\partial\,\varphi(t)}{\partial t}$

Im 2D Vektorfeld:

Winkelgeschwindigkeit $\omega = \dfrac{1}{2}\,\mathrm{rot}\,\vec{v}(\vec{x})$

Rotation starrer Körper

- *Rotation in 3D:*



Ein Körper dreht sich in der Zeit t um den Winkel φ um die Achse $\vec{h}$.

Der Körper dreht sich mit der Winkelgeschwindigkeit ω um die Achse $\vec{h}$.

Die Rotation $\mathrm{rot}\,\vec{v}(\vec{x})$ für 3D Vektorfelder ist kein Skalar, sondern ein Vektor

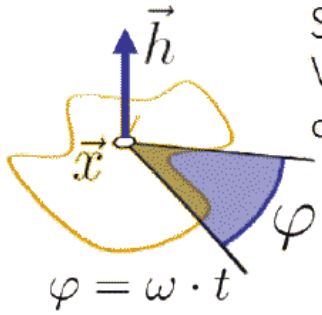
Rotation in 3D

- Betrachte die 2D Rotation in den einzelnen Ebenen:

$$\mathbf{J}_{\vec{v}}(\vec{x}) = \begin{pmatrix} \frac{\partial v_x}{\partial x} & \frac{\partial v_x}{\partial y} & \frac{\partial v_x}{\partial z} \\ \frac{\partial v_y}{\partial x} & \frac{\partial v_y}{\partial y} & \frac{\partial v_y}{\partial z} \\ \frac{\partial v_z}{\partial x} & \frac{\partial v_z}{\partial y} & \frac{\partial v_z}{\partial z} \end{pmatrix}$$

$$\mathrm{rot}\,\vec{v}(\vec{x}) = \begin{pmatrix} \frac{\partial v_z}{\partial y} - \frac{\partial v_y}{\partial z} \\ \frac{\partial v_x}{\partial z} - \frac{\partial v_z}{\partial x} \\ \frac{\partial v_y}{\partial x} - \frac{\partial v_x}{\partial y} \end{pmatrix}$$

- *Interpretation:* An der Stelle $\vec{x}$ dreht sich die Strömung lokal mit der Winkelgeschwindigkeit ω um die Achse $\vec{h}$:



$$\vec{\omega} = \frac{1}{2}\,\mathrm{rot}\,\vec{v}(\vec{x})$$

Winkelgeschwindigkeit:

$$\omega = \|\vec{\omega}\|$$

Rotationsachse:

$$\vec{h} = \frac{\vec{\omega}}{\|\vec{\omega}\|}$$

- Wie stark ist die lokale Rotation der Strömung um eine *bestimmte gegebene Achse* $\vec{r}$?

$$\mathrm{rot}_{\vec{r}}\,\vec{v} = \frac{\langle \mathrm{rot}\,\vec{v} \circ \vec{r} \rangle}{\|\vec{r}\|}$$

(Projektion des Rotationsvektors auf die Achse)

- *Spezialfall „streamwise rotation":*
  Lokale Rotation um die Strömungsrichtung:

$$\omega_s = \frac{\langle \mathrm{rot}\,\vec{v} \circ \vec{v} \rangle}{\|\vec{v}\|}$$

Ableitungen in 3D

$\frac{\partial}{\partial x}$ ← Differentialoperator (partielle Ableitung in x-Richtung)

Angewendet auf eine Funktion $f(x, y, z)$ liefert er die partielle Ableitung in x-Richtung:

$$\frac{\partial}{\partial x} f(x, y, z)$$

- Jacobi-Matrix, Divergenz und Rotation können mit Hilfe des *Nabla-Operators* ausgedrückt werden

$$\nabla = \begin{pmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{pmatrix}$$

Divergenz

$$\begin{aligned} \operatorname{div} \vec{v} &= \langle \nabla \circ \vec{v} \rangle \\ &= \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z} \end{aligned}$$

Rotation:

$$\operatorname{rot} \vec{v}(\vec{x}) = \nabla \times \vec{v}(\vec{x}) = \begin{pmatrix} \frac{\partial v_z}{\partial y} - \frac{\partial v_y}{\partial z} \\ \frac{\partial v_x}{\partial z} - \frac{\partial v_z}{\partial x} \\ \frac{\partial v_y}{\partial x} - \frac{\partial v_x}{\partial y} \end{pmatrix}$$

Jacobi-Matrix:

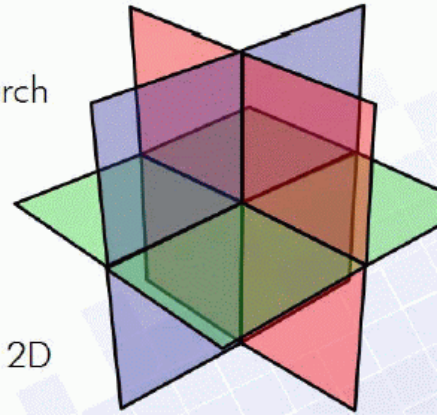
$$\mathbf{J}_{\vec{v}} = \vec{v} \cdot \nabla^T = \begin{pmatrix} \frac{\partial v_x}{\partial x} & \frac{\partial v_x}{\partial y} & \frac{\partial v_x}{\partial z} \\ \frac{\partial v_y}{\partial x} & \frac{\partial v_y}{\partial y} & \frac{\partial v_y}{\partial z} \\ \frac{\partial v_z}{\partial x} & \frac{\partial v_z}{\partial y} & \frac{\partial v_z}{\partial z} \end{pmatrix}$$

Kritische Punkte in 3D

- Bestimmung und Klassifikation analog zu 2D: Eigenwerte und Eigenvektoren der Jacobi-Matrix

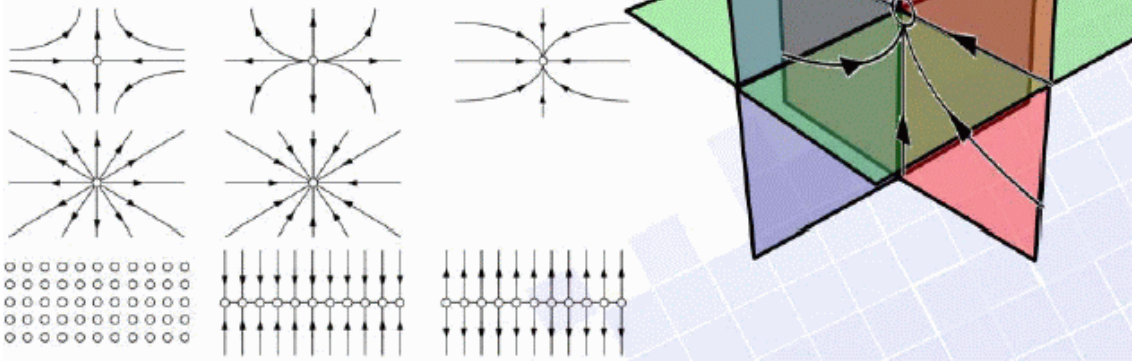
Prinzip:

- Betrachte die Ebenen, die durch Eigenvektoren aufgespannt werden
- Betrachte die Vektoren auf die Ebene projiziert
- Klassifiziere die Ebene wie in 2D



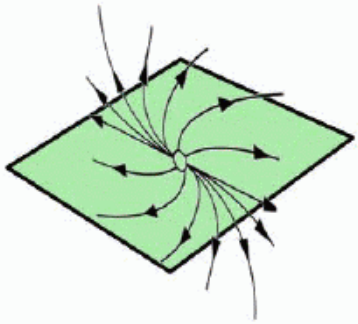
3 reelle Eigenvektoren:

Betrachte Ebenen, die durch je zwei Eigenvektoren aufgespannt werden.

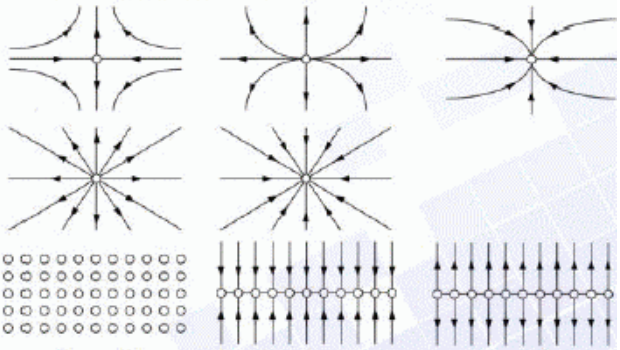


2 reelle Eigenvektoren:

Betrachte die Ebene, die durch die Eigenvektoren aufgespannt wird.



in der Eigenvektor-Ebene:



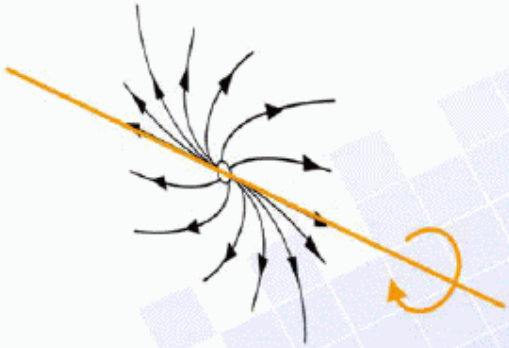
in allen Ebene die koplanar mit einem der Eigenvektoren sind:

1 reeller Eigenvektor:

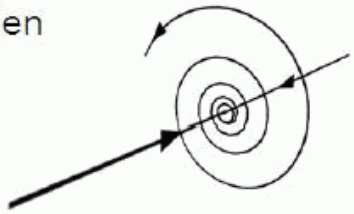
Betrachte die Gerade, die durch den Eigenvektor aufgespannt wird.

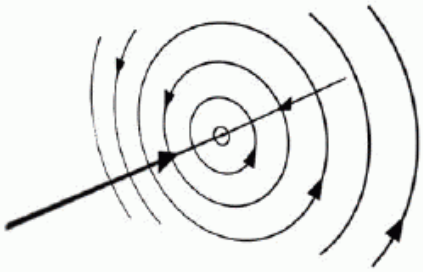
in allen Ebenen, die koplanar mit dem Eigenvektor sind zu:

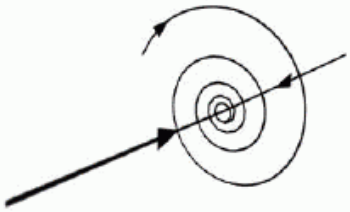
Beispiel:



Komplexe Eigenwerte treten nur als Paar konjugiert komplexer Zahlen auf. *Daher existiert bei einer 3x3 Matrix immer mindestens ein reeller Eigenwert!*







Je nach Vorzeichen des reellen Eigenwerts führt die Strömung zu dem kritischen Punk hin oder weg.

(Rezk-Salama, o.J.)

Auffinden der kritischen Punkte

Bestimmung kritischer Punkte - Gitter:

- Für jede Gitterzelle *C*
  - Finde Stellen x_i mit $v(x_i)=0$
  - Für jedes x_i prüfe, ob $x_i \in C$, wenn ja wurde kritischer Punkt gefunden
- Für alle kritischen Punkte
  - Analysiere Jakobimatrix
  - Initiiere Streamlines in Hauptrichtungen

Bestimmung kritischer Punkte – Simplices:

Dimension: *d*

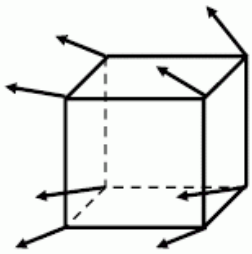
$$\sum_{i=0}^{d+1} \alpha_i \cdot \mathbf{v}_i = \mathbf{0} \qquad \sum_{i=0}^{d+1} \alpha_i = 1 \quad \rightarrow \quad \begin{pmatrix} \mathbf{v}_0 & \cdots & \mathbf{v}_d \\ 1 & \cdots & 1 \end{pmatrix} \cdot \begin{pmatrix} \alpha_0 \\ \vdots \\ \alpha_{d-1} \\ \alpha_d \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}$$

Lösung eines $(d+1)\times(d+1)$ linearen Gleichungssystems

3d Intervallschachtelung:

Beobachtung:

- Sind alle x/y/z-Komponenten der $\mathbf{v}_{0...7}$ rein positiv/negativ, gibt es keinen kritischen Punkt



Rekursiver Algorithmus:

`if` kein kritischer Punkt möglich `return false`
`if` maximale Rekursionstiefe `return true`
`recursion` für 8 Unterzellen

(Bartz 2005)

Visualisierungsverfahren

Direkte Verfahren

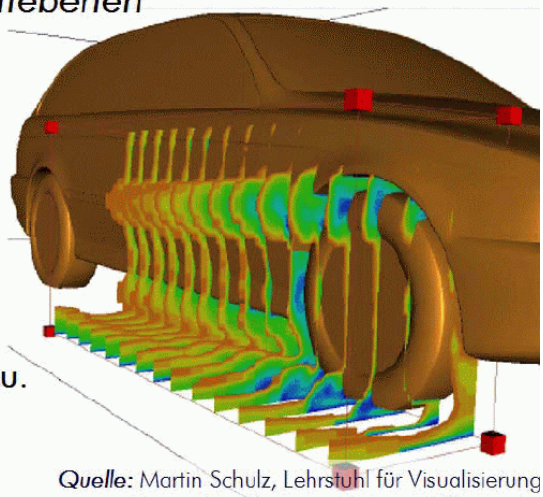
- Farbkodierung
  - auf Schnittebenen
  - im Volumen (Volume Rendering)
  - Isoflächen
- Vektorpfeile
- Glyphen/Icons

Integrationsbasierte Verfahren

Farbkodierung

3D Skalarfelder z.B. Dichte, Druck, Geschwindigkeitsbetrag

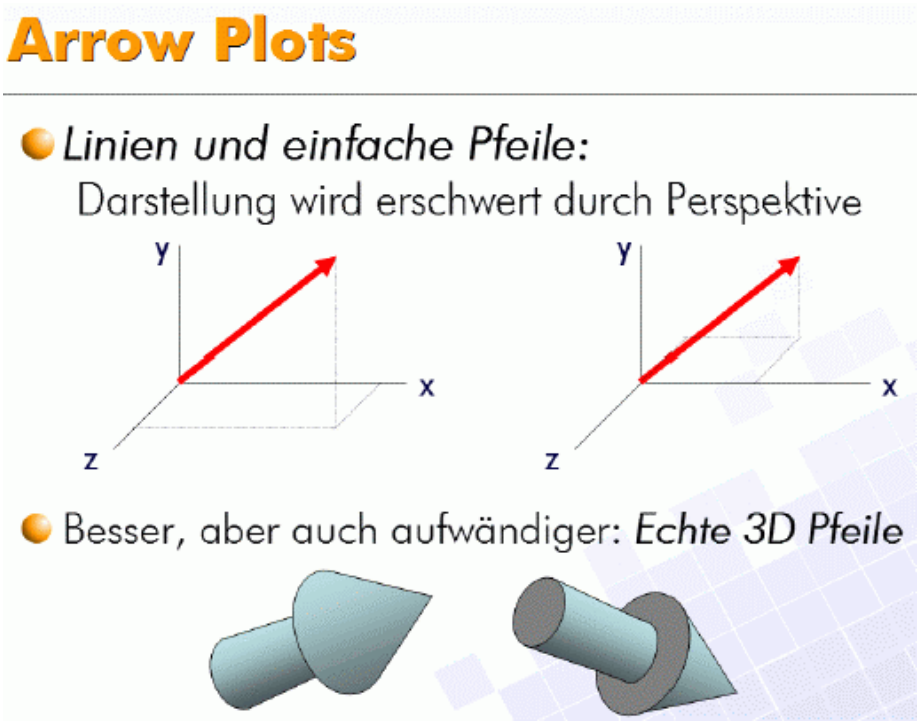
Abgebildet auf Schnittebenen



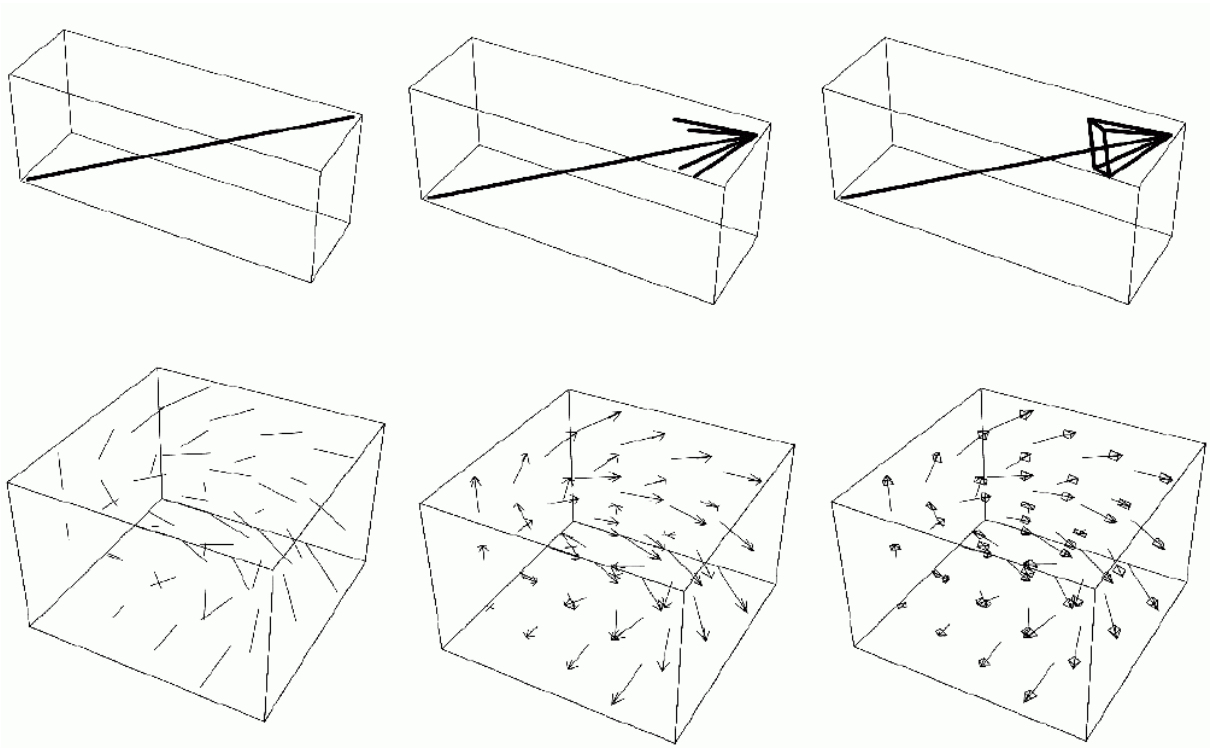
Transparenz/Opazität kommt als zusätzliche Farbkomponente hinzu.

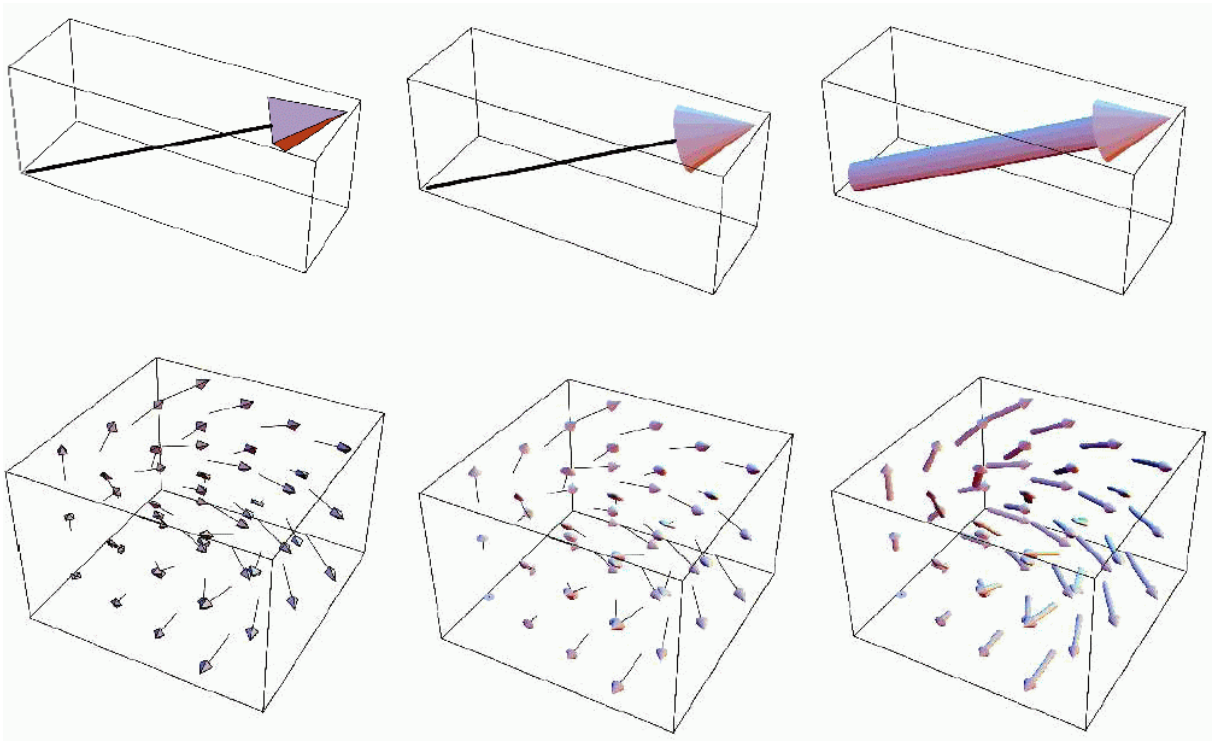
Quelle: Martin Schulz, Lehrstuhl für Visualisierung und Interaktive Systeme, Universität Stuttgart

Darstellung von 3D-Skalarfeldern im Volumen:
siehe nächstes Kapitel, "Volumenrendering"
(z.B. mit Isoflächen)



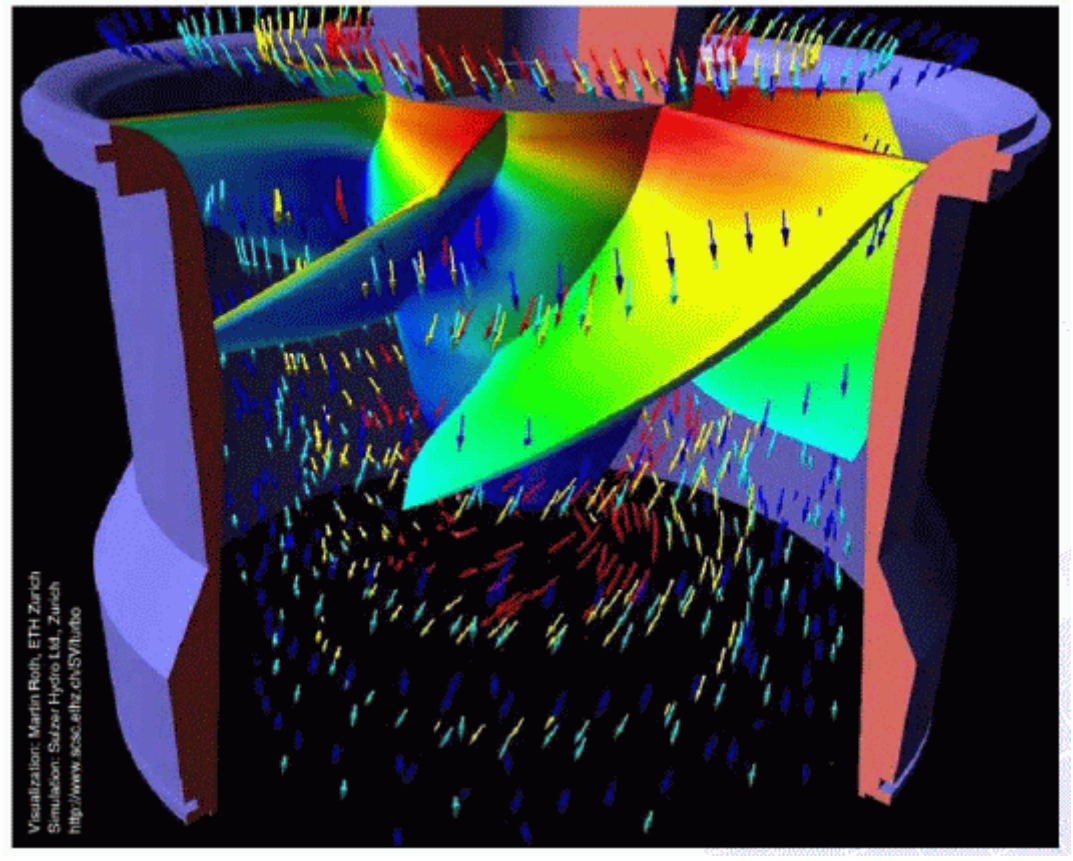
(Rezk-Salama, o.J.)





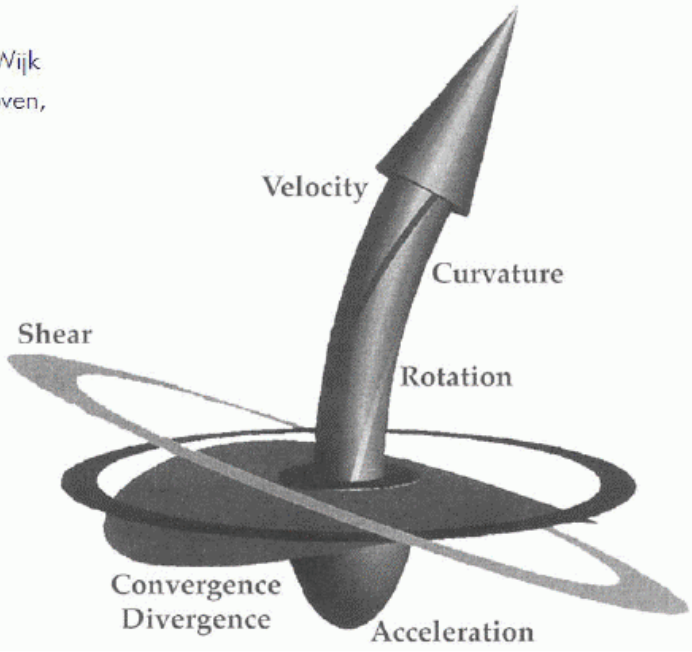
(Weimar 2005)

Beispiel: Arrow Plots in 3D



Glyphen/Icons in 3D

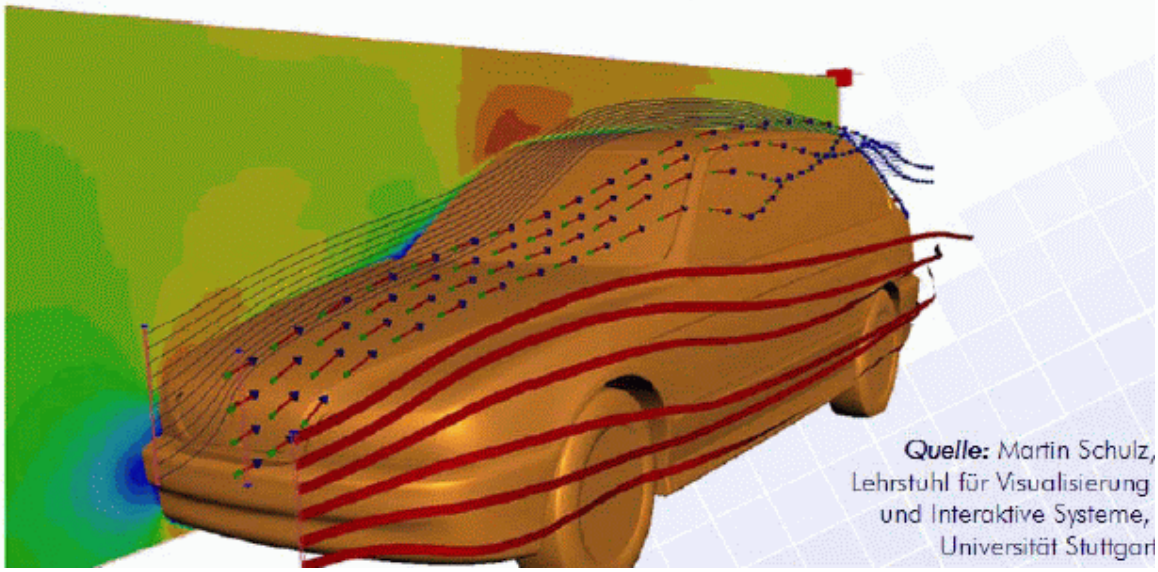
Quelle:
de Leeuw and van Wijk
University of Eindhoven,
the Netherlands



Integrationsbasierte Verfahren

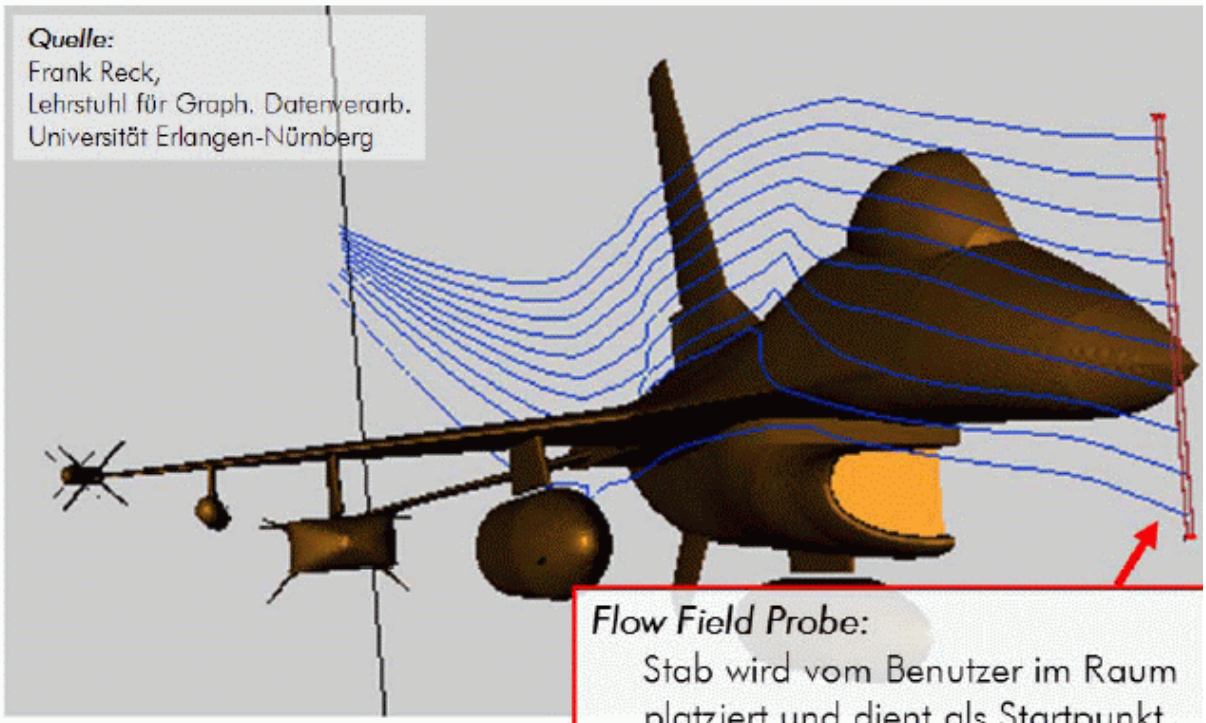
- **Linienbasierte Techniken:**

Wie bisher: Partikelbahnen (analog 2D),
Streamlines, Streaklines, Pathlines



Quelle: Martin Schulz,
Lehrstuhl für Visualisierung
und Interaktive Systeme,
Universität Stuttgart

Partikelbahnen



Flow Field Probe:
Stab wird vom Benutzer im Raum platziert und dient als Startpunkt für Partikelbahnen.

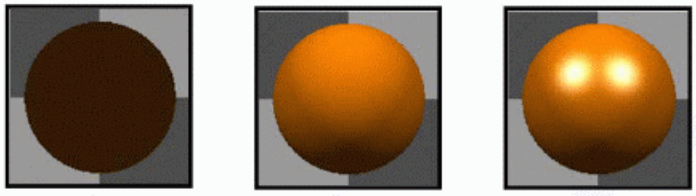
Problem: Bei 3D Partikelbahnen, die als einfache Linien dargestellt werden, fehlt der „räumliche Tiefeneindruck"

Möglichkeiten:

- *Beleuchtungseffekte:*
  Räumliche Tiefe kann sehr gut durch *lokale Beleuchtung* erzeugt werden.
  - Erzeuge echte 3D Geometrie:
    Streamballs, Stream Ribbons, Stream Tubes
  - Beleuchtungsverfahren für Linien:
    Beleuchtete Partikelbahnen

Erinnerung:

Blinn-Phong Beleuchtungsmodell:



ambient | diffuse (Lambert) | spekular (spiegelnd)

$$I = I_a + I_d \langle \vec{n} \circ \vec{l} \rangle + I_s \langle \vec{n} \circ \vec{h} \rangle^r$$

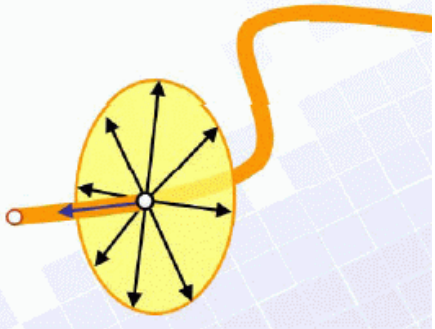
- *Für Partikelbahnen:*
  Was ist die Oberflächennormale einer Linie?

Beleuchtete Partikelbahnen

Was nehme ich als Normalenvektor der Linie?

1. Interpretiere die Partikelbahn als eine *sehr dünne Röhre*:

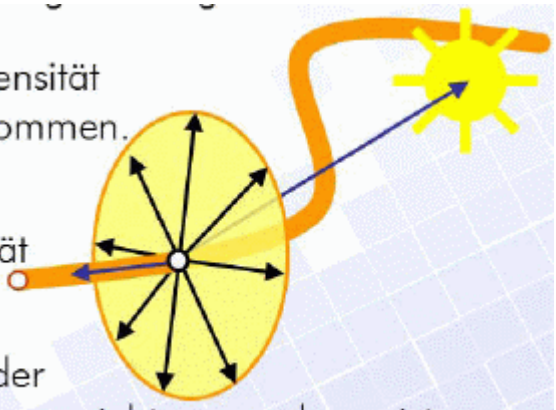
   Die Normalenvektoren an einem Punkt sind alle Vektoren, die senkrecht auf die Strömungsrichtung stehen.



2. Die hellste Beleuchtungsintensität wird am stärksten wahrgenommen.

Wähle den Vektor, der die hellste Beleuchtungsintensität erzeugt.

Das ist der Vektor, der mit der Lichtrichtung und der Strömungsrichtung coplanar ist.

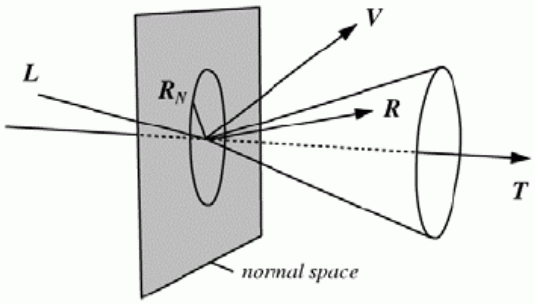


(Rezk-Salama, o.J.)

Beleuchtung von Linien:

Modell: Linie ist infinitesimal dünner Zylinder

➔ Kegel von Reflektionsvektoren



➔ Wähle **R** in Ebene von **L** und **T**

(Bartz 2005)

$$\langle \vec{n} \circ \vec{l} \rangle = \sqrt{1 - \langle \vec{t} \circ \vec{l} \rangle^2}$$

- Diffuser Term:

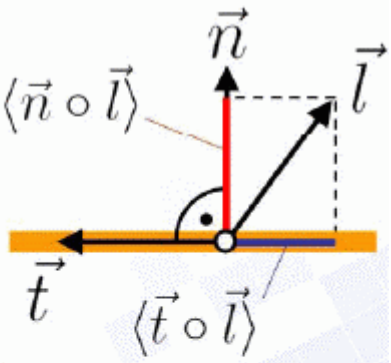
$$I_d \langle \vec{n} \circ \vec{l} \rangle = I_d \sqrt{1 - \langle \vec{t} \circ \vec{l} \rangle^2}$$

- Spekularer Term (analog):

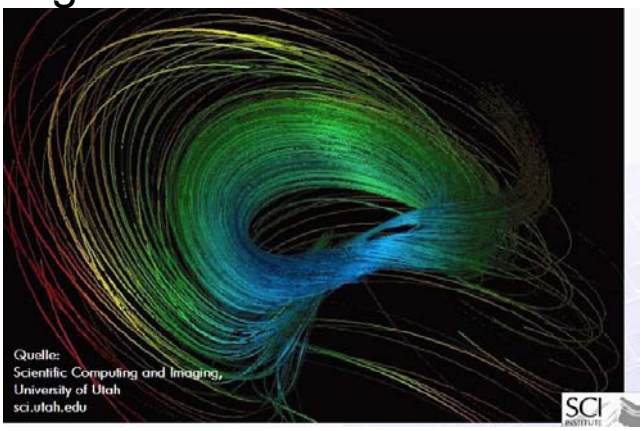
$$I_s \langle \vec{n} \circ \vec{h} \rangle^r = I_s \sqrt{1 - \langle \vec{t} \circ \vec{h} \rangle^2}^r$$

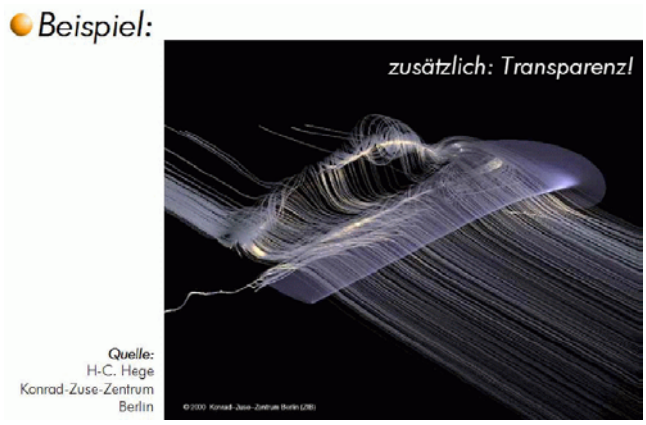
- *Blinn-Phong Modell für Linien:*

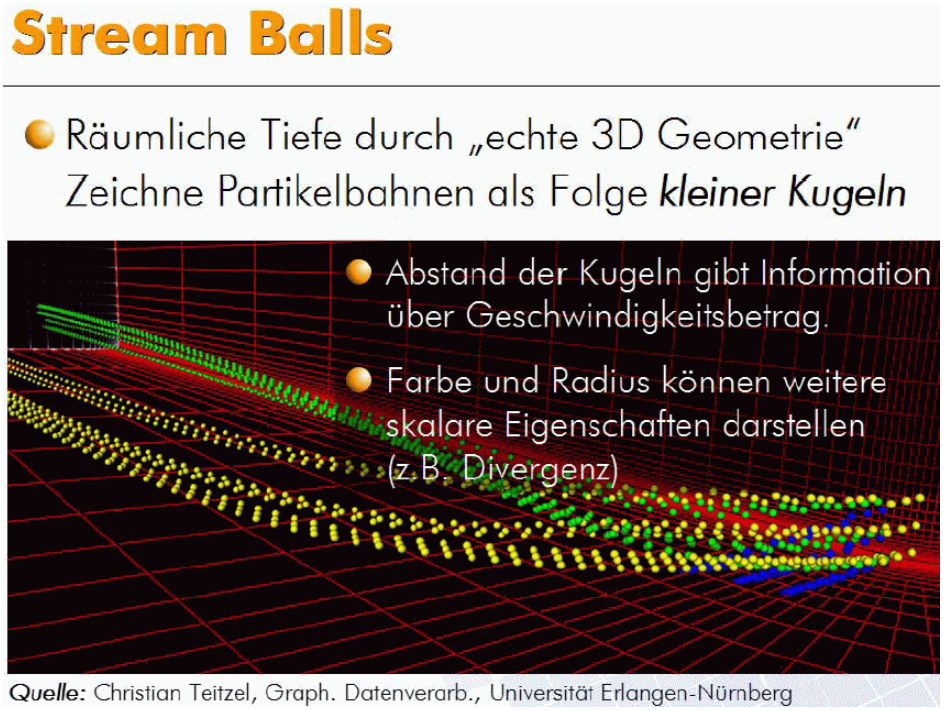
$$I = I_a + I_d \sqrt{1 - \langle \vec{t} \circ \vec{l} \rangle^2} + I_s \sqrt{1 - \langle \vec{t} \circ \vec{h} \rangle^2}^r$$



Ergebnisse:


Quelle:
Scientific Computing and Imaging,
University of Utah
sci.utah.edu
SCI

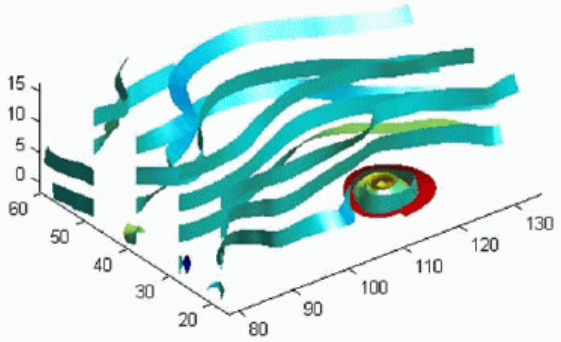

Beispiel:
zusätzlich: Transparenz!
Quelle:
H-C. Hege
Konrad-Zuse-Zentrum
Berlin


Stream Balls
Räumliche Tiefe durch „echte 3D Geometrie"
Zeichne Partikelbahnen als Folge kleiner Kugeln
Abstand der Kugeln gibt Information über Geschwindigkeitsbetrag.
Farbe und Radius können weitere skalare Eigenschaften darstellen (z.B. Divergenz)
Quelle: Christian Teitzel, Graph. Datenverarb., Universität Erlangen-Nürnberg


Kugeln können miteinander verschmelzen (Meta-Balls)
und komplexere Objekte (Röhren, Flächen) bilden.
1.5
0.75
0
Quelle: Hans Hagen, IEEE Visualization 1994

Strömungsbänder

- *Idee:* Zeige die lokale Rotation der Strömung durch Darstellung eines Bandes *(stream ribbon)* mit bestimmter Breite.

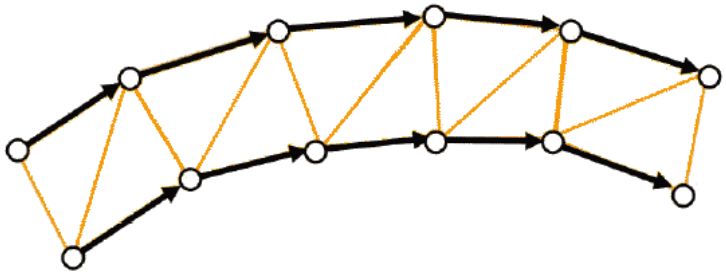


Streamribbins mit MATHLAB
Quelle: www.mathworks.com

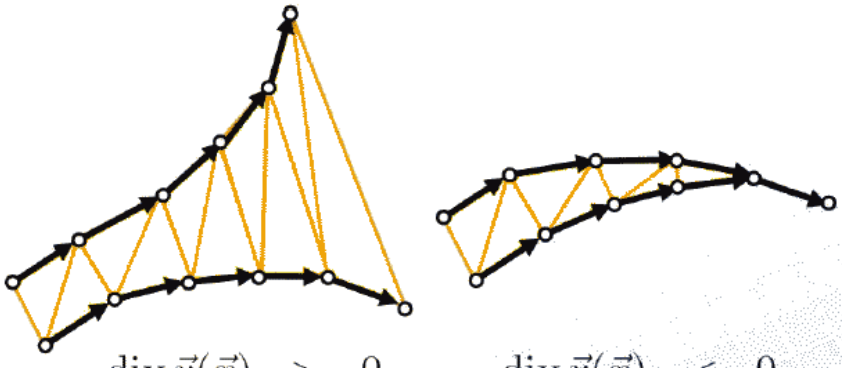
Erzeugen von Strömungsbändern

Methode 1: Zwei Partikelbahnen.

- Wähle zwei Startpunkte, die nahe beieinander liegen.
- Bestimme zwei Partikelbahnen
- Verbinde die beiden Bahnen durch Polygone (Dreiecke)



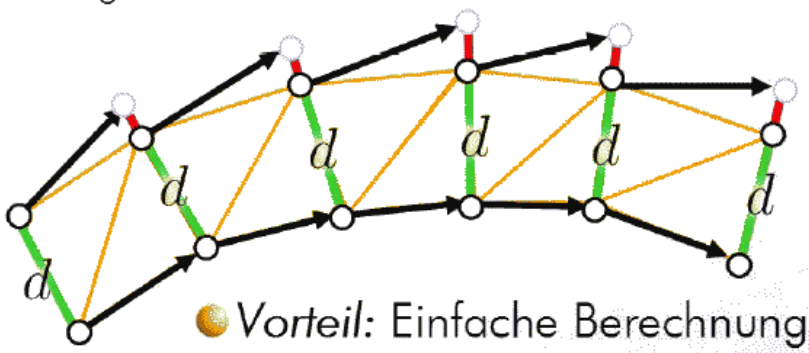
- *Vorteil:* Einfache Berechnung
- *Nachteil:* Gutes Ergebnis nur bei sehr geringer Divergenz.



$\operatorname{div} \vec{v}(\vec{x}) > 0$ $\operatorname{div} \vec{v}(\vec{x}) < 0$

Methode 2: Methode 1 mit Korrektur.

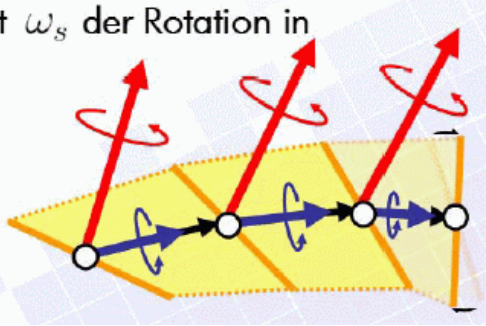
- Wähle zwei Startpunkte, in bestimmtem Abstand d.
- Bestimme zwei Folgepunkte
- Korrigiere den Abstand durch Verschieben eines Punktes

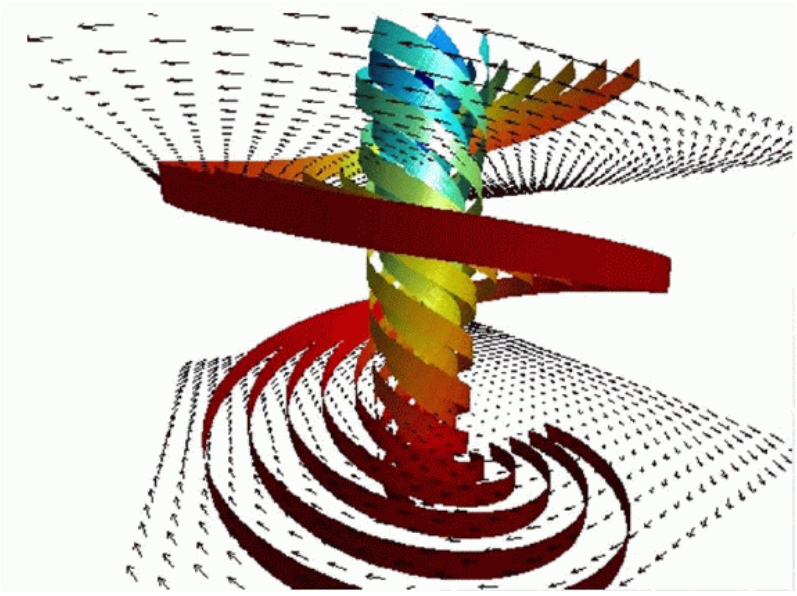


- *Vorteil:* Einfache Berechnung
- *Nachteil:* „Nur eine Kante stimmt"

Methode 3: Direkte Bestimmung der Rotation

- Bestimme eine einzelne Partikelbahn.
- Für jeden Punkt
  - Bestimme den Rotationsvektor *(durch Zent. Diff. + Interp.)*
  - Projiziere den Rotationsvektor auf die Strömungsrichtung
    = **Winkelgeschwindigkeit ω_s der Rotation in Strömungsrichtung**
  - Lege eine Kante in den Startpunkt
  - Drehe die Kante bei jedem Zeitschritt τ um den Winkel $\varphi = \omega_s \cdot \tau$



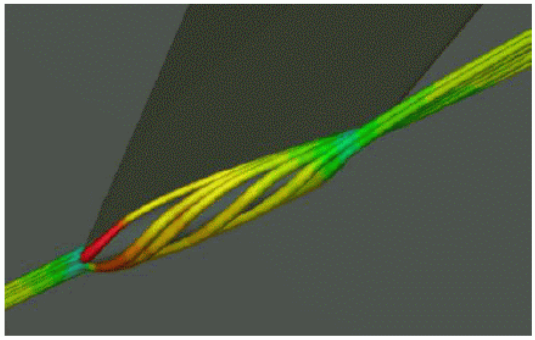


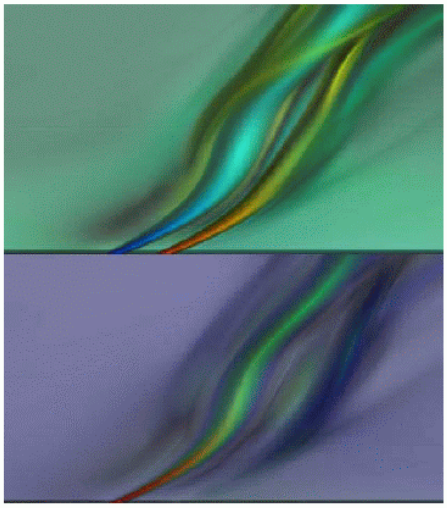
(Rezk-Salama, o.J.)

Flächendarstellung

Streamtubes/Stromröhren

- Kontur wird durch Feld verfolgt





(Bartz 2005)

Stream Tubes

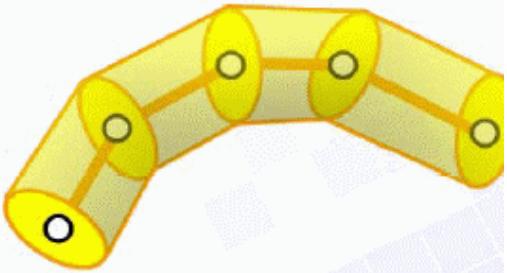
- *Strömungsröhren:*

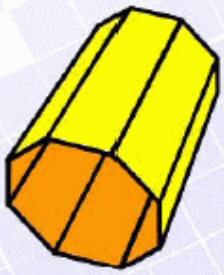
  Einfache Variante: Zeiche die Partikelbahn als Röhre mit festem Radius.

  Bestimme eine Partikelbahn.

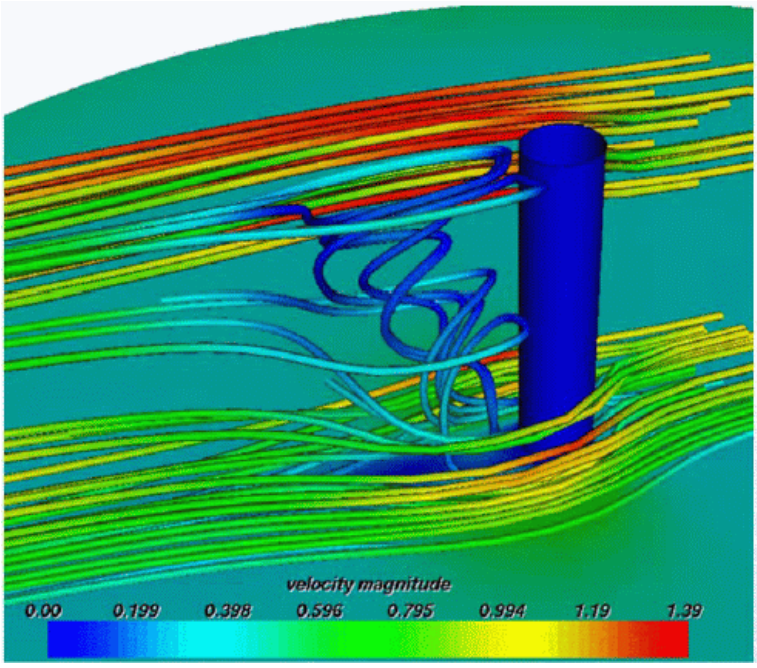
  Für jeden Zeitschritt:

  - Bestimme einen Kreis mit festem Radius, der senkrecht auf der Strömungsrichtung steht.
  - Verbinde die Kreise durch zylinderförmige Flächen.





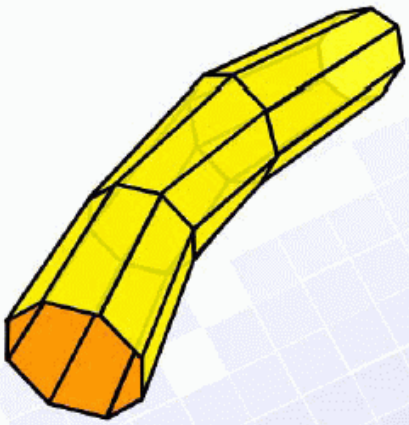
Die Zylinderflächen werden durch Polygone approximiert und gezeichnet.



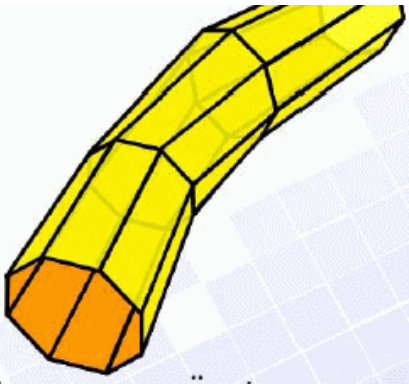
- *Strömungsröhren:*

  Aufwändigere Variante: Zeiche eine Röhre, die aus mehreren Partikelbahnen besteht.

  Wähle einen Kreis.
  Für jeden Punkt des Kreises:
  - Bestimme eine Partikelbahn.
  - Verbinde die Kreise aufeinanderfolgender Zeitschritte durch zylinderförmige Flächen.



Physikalische Eigenschaften dieser Variante:

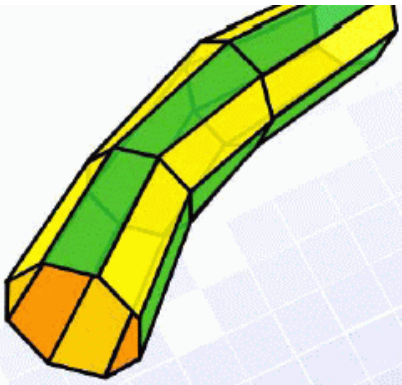


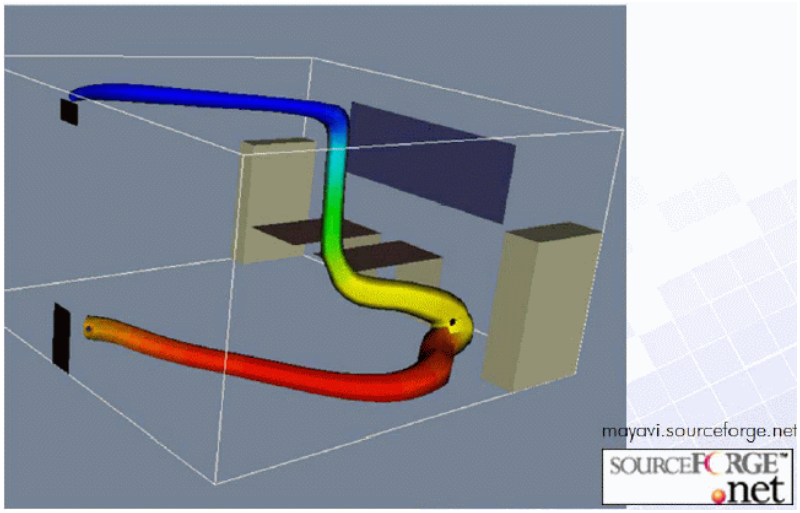
- Die „Wand" der Röhre ist *impermeabel* (=undurchdringlich): Partikel innerhalb der Röhre verlassen die Röhre niemals!
- *Bei inkompressibler Strömung:* Aufgrund der Massenerhaltung bedeutet eine Änderung des Durchmessers der Röhre eine lokale Änderung der Geschwindigkeit.

Durch **Einfärben der einzelnen Streifen** kann die lokale **Rotation** in Strömungsrichtung

$$\omega_s = \frac{\langle \mathrm{rot}\, \vec{v} \circ \vec{v} \rangle}{\|\vec{v}\|}$$

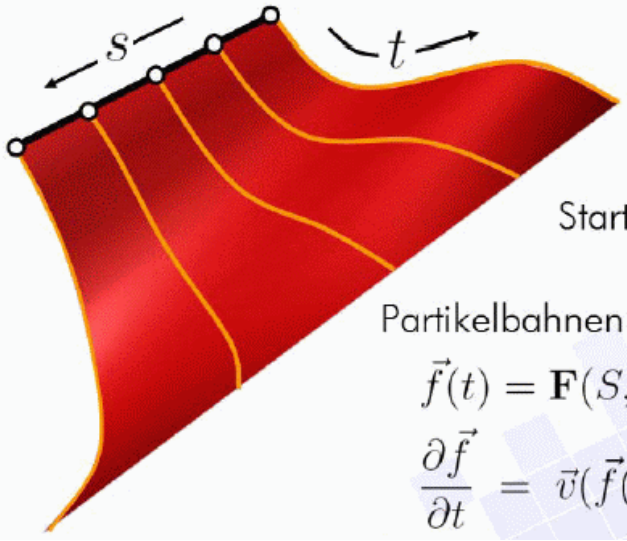
visualisiert werden





Stream Surfaces

- *Definition: Strömungsflächen* sind Flächen die sich aus einzelnen Partikelbahnen zusammensetzen.



Als parametrische Fläche:

$$\mathbf{F}(s,t)$$

Startkurve (Time Line):

$$\vec{g}(s) = \mathbf{F}(s,0)$$

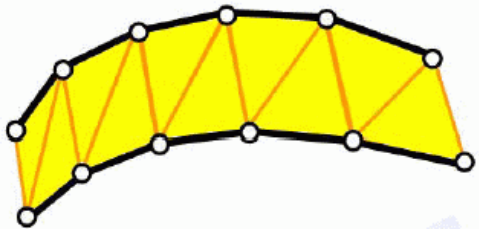
Partikelbahnen:

$$\vec{f}(t) = \mathbf{F}(S,t) \quad \text{mit} \quad S = \text{const.}$$

$$\frac{\partial \vec{f}}{\partial t} = \vec{v}(\vec{f}(t))$$

Bestimmung der Strömungsfläche.

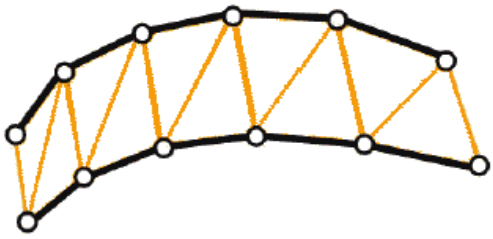
1. Wähle eine *Startkurve* (als Linienzug)
2. Verschiebe jeden Vertex um einen *Zeitschritt*
3. *Trianguliere* die Zwischenfläche.



Regelmäßige Triangulierung

Einfaches Verfahren:

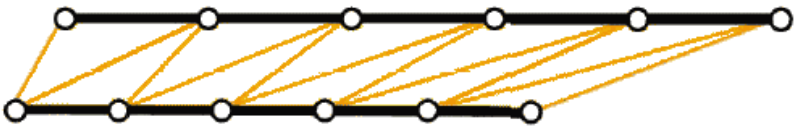
- Füge die Kanten ein, die den Partikelbahnen entsprechen
- Teile die entstehenden Vierecke in regelmäßige Dreiecke.



Vorteile:

- Schnell und unkompliziert.
- Es entsteht direkt ein *Triangle Strip*, der effizient gerendert werden kann (z.B. `GL_TRIANGLE_STRIP` in OpenGL)

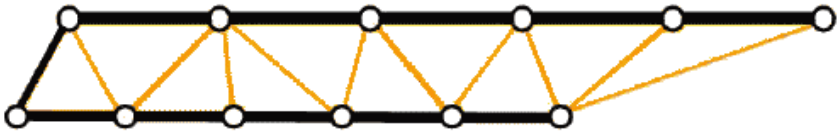
Problematik:



- Durch *Scherung* im Vektorfeld (d.h. starke Beschleunigung senkrecht zur Strömungsrichtung) können sehr lange spitze Dreiecke entstehen.

Minimale Triangulierung

Triangulierung, die die Kantenlängen des Streifens minimiert *(Greedy Algorithm)*.



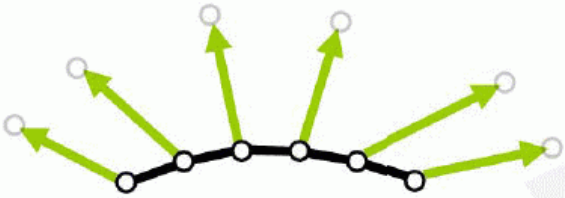
Betrachte die nächsten Punkte auf den Linien:

- Wähle von den möglichen Folgekanten jeweils die kürzere

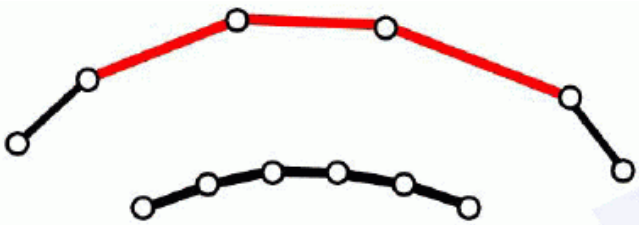
Vorteil:

- Gute Triangulierung der Streifen.
- Einfach zu berechnen

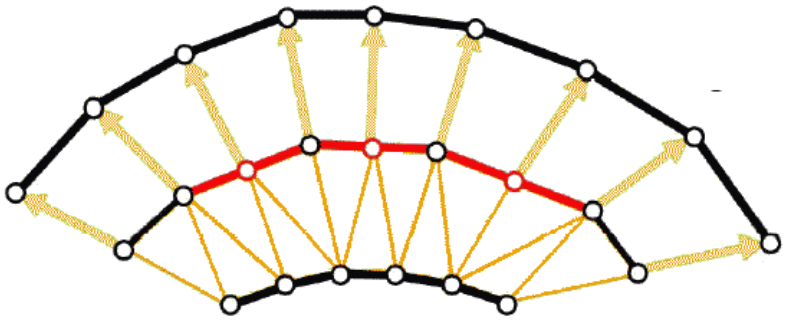
- *Problematik: Divergenz* $\operatorname{div} \vec{v} > 0$



Problem: Die Kanten werden immer länger.
Lösungsansatz:



- *Splitting:* Unterteile die Kanten der Führungskurve, wenn sie eine bestimmte Länge überschreiten.

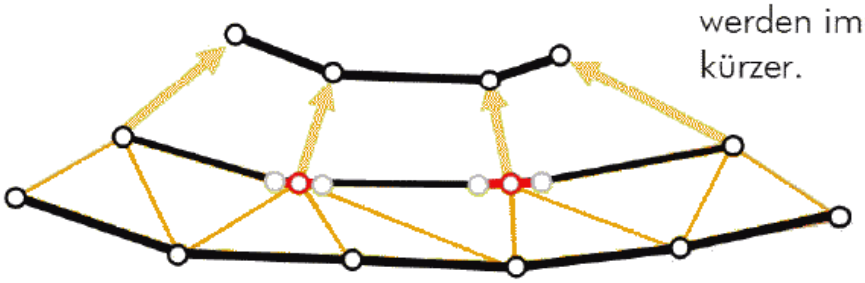


- *Splitting:* Unterteile die Kanten der Führungskurve, wenn sie eine bestimmte Länge überschreiten.

  Anmerkung: Regelmäßige Triangulierung ist nicht mehr möglich, da die Anzahl der Vertices auf den Time Lines unterschiedlich ist!

- *Problematik: Konvergenz* $\operatorname{div} \vec{v} < 0$

Problem: Die Kanten werden immer kürzer.

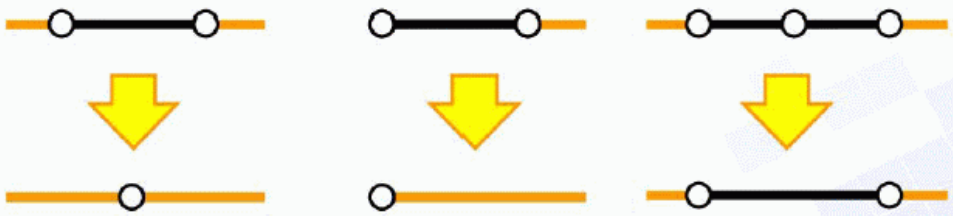


- *Merging:* Kollabiere Kanten der Führungskurve, wenn sie eine bestimmte Länge unterschreiten.

Merging

- Edge Collapse: Kollabieren einer Kante

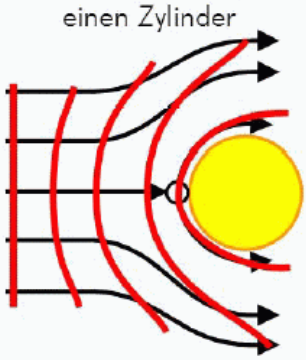
Reguläre Kante *Randkante* *Mehrfach-Kollaps*



- Nach dem *Merging* darf die Führungslinie keine zu kurzen Kanten mehr enthalten!
- Bei sehr starker Konvergenz müssen Kanten rekursiv kollabiert werden!

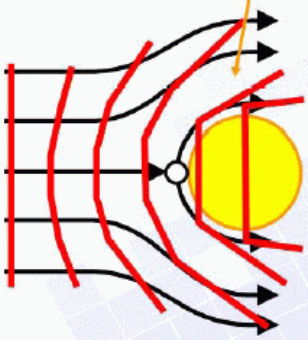
Problematik: Hindernisse

Beispiel: Strömung um einen Zylinder



Idealer Fall

Strömunsfläche durchdringt das Objekt



Realer Fall

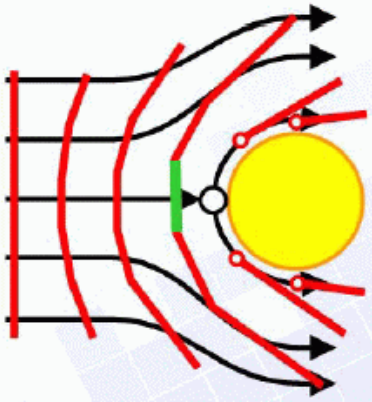
Grund: Diskretisierung

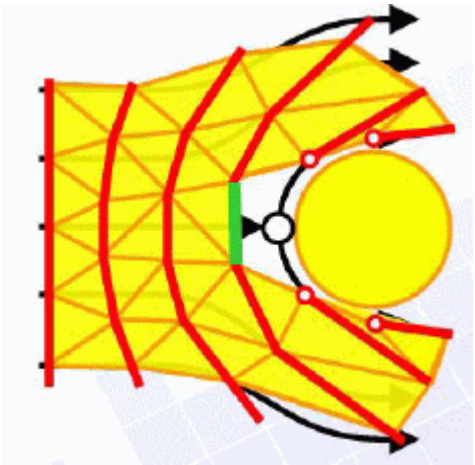
Ripping:

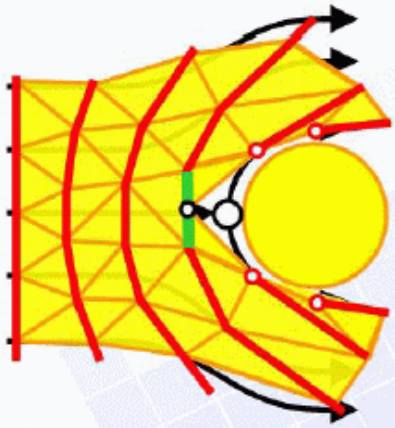
Zerreiße die Strömungsfläche, wenn die Punkte einzelner Liniensegmente sehr stark in unterschiedliche Richtungen gezogen werden.

Zerreiße die Kante, wenn

$$\|\vec{v}_a\| > v_{\max}$$
$$\|\vec{v}_b\| > v_{\max}$$
$$\angle(\vec{v}_a\vec{v}_b) > \varphi_{\max}$$







- *„Kosmetik"*: Füge den Mittelpunkt der entfernten Kante und zwei Dreiecke ein

Stream Surfaces

Bestimmung der Strömungsfläche.

1. Wähle eine *Startkurve* (als Linienzug)
2. Verschiebe jeden Vertex um einen *Zeitschritt*
   - *Ripping:* Entferne Kanten die „zu schnell wachsen".
   - *Splitting:* Teile Kanten, die zu lang geworden sind.
   - *Merging:* Kollabiere Kanten die zu kurz geworden sind.
3. *Trianguliere* die Zwischenfläche.

- *Frage:* Ist jede beliebige Linie als Startkurve geeignet?

Bestimmung der Strömungsfläche.

1. Wähle eine *Startkurve* (als Linienzug)

Eine Strömungslinie selbst ist beispielsweise keine gute Startkurve:

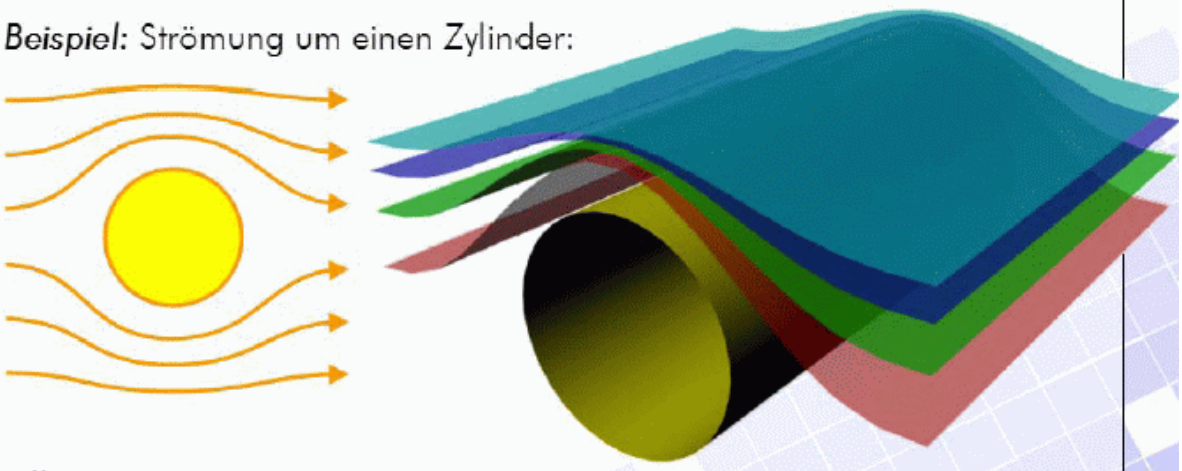
Auch Kurven, die annähernd in Strömungsrichtung verlaufen sind weniger gut:

- *Antwort:* Nein.

  Die Startkurve sollte möglichst orthogonal zur Strömung verlaufen.

- *Idee:* Untersuche die Strömungsflächen, die man bei einfachen Strömungen „intuitiv wählen würde".

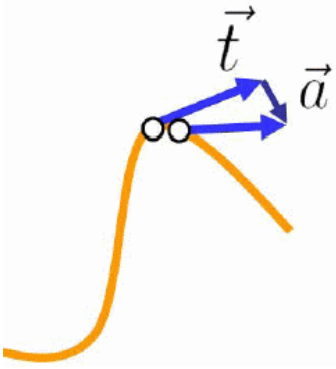
Beispiel: Strömung um einen Zylinder:



Übertrage die erkannten Prinzipien auf komplexe Strömungen.

Frenet Frame

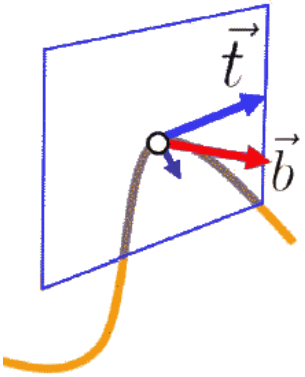
- Betrachte den *Frenet Frame (Dreibein)* einer Strömungslinie:



- *Die Beschleinigung*

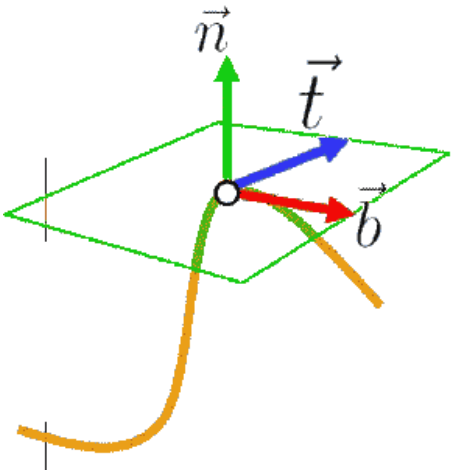
$$\vec{a} = \frac{\partial \vec{v}}{\partial t}$$

beschreibt die lokale Änderung der Geschwindigkeit und somit die *Krümmung* der Kurve



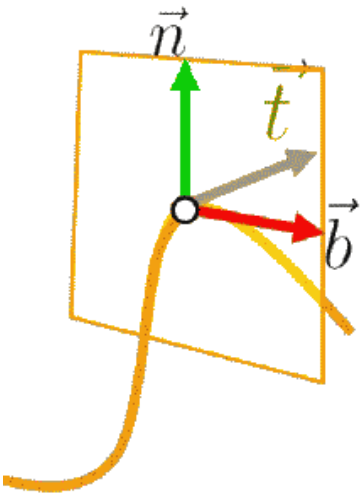
- Die Tangente und der Beschleunigungsvektor spannen eine *Schmiegebene* auf.
- Die Kurve verläuft *lokal* innerhalb dieser Ebene.
- *Die Binormale* ist der Normalenvektor dieser Schmiegebene

$$\vec{b} = \frac{\vec{t} \times \vec{a}}{\| \vec{t} \times \vec{a} \|}$$



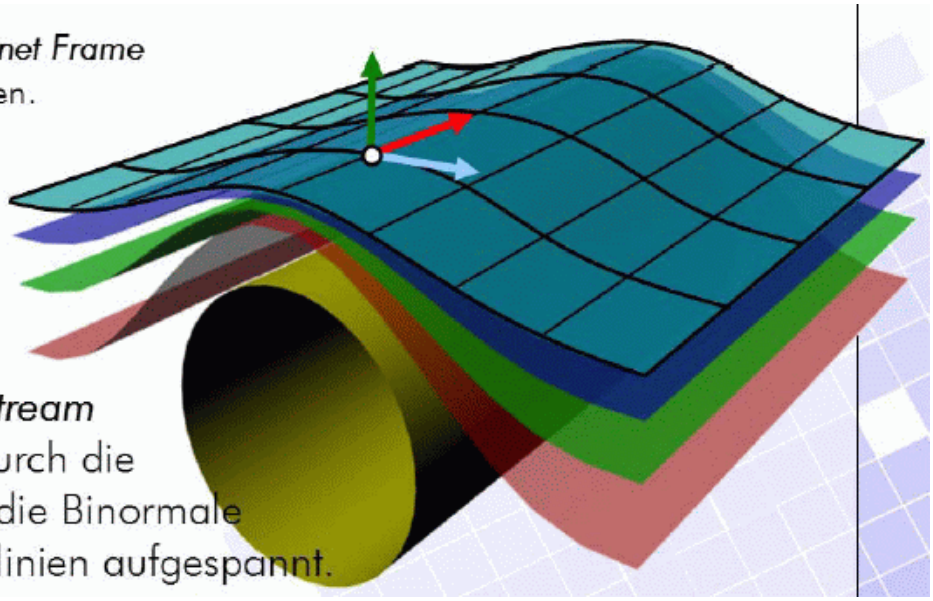
- Die Tangente und die Binormale spannen die *rektifizierende Ebene* auf.
- *Die Normale der Kurve* ist der Normalenvektor dieser rektifizierenden Ebene:

$$\vec{n} = \vec{b} \times \vec{v}$$



- Die Normale und die Binormale spannen die *Normalebene* auf.
- *Die Tangente der Kurve* ist der Normalenvektor dieser Normalebene.

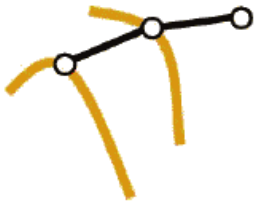
Betrachte den *Frenet Frame* der Strömungslinien.



Die *Principal Stream Surface* wird durch die Tangente und die Binormale der Strömungslinien aufgespannt.

Bestimmung der Startkurve

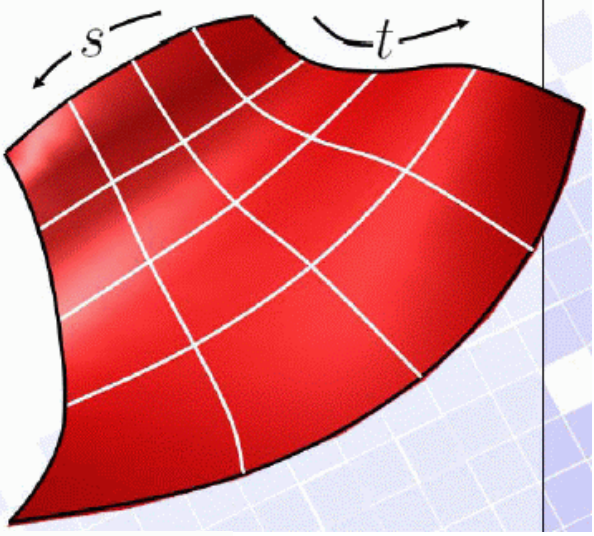
- Beginne mit einem Punkt.
- Bestimme den ***Frenet Frame*** an die Strömungslinie in diesem Punkt
- Gehe einen kleinen Schritt in Richtung der ***Binormalen***



Die Startkurve der Principal Stream Surface ist eine *Integralkurve in Richtung der Binormalen.*

- Strömungsflächen zeigen nicht die Richtung der Strömung innerhalb der Fläche.

Verwende *Texturen* um die Strömung innerhalb der Strömungsfläche darzustellen.

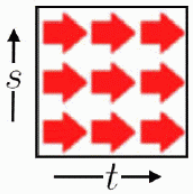


Texturkoordinaten (u,v) können direkt aus der Parametrisierung der Fläche $\mathbf{F}(s,t)$ abgeleitet werden.

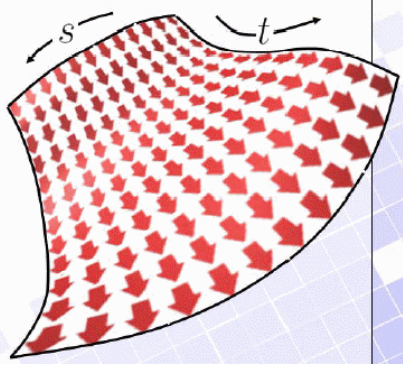
Texturkoordinate $u = s$*: Time Lines*
Texturkoordinate $v = t$*: Path Lines*

Texturierung

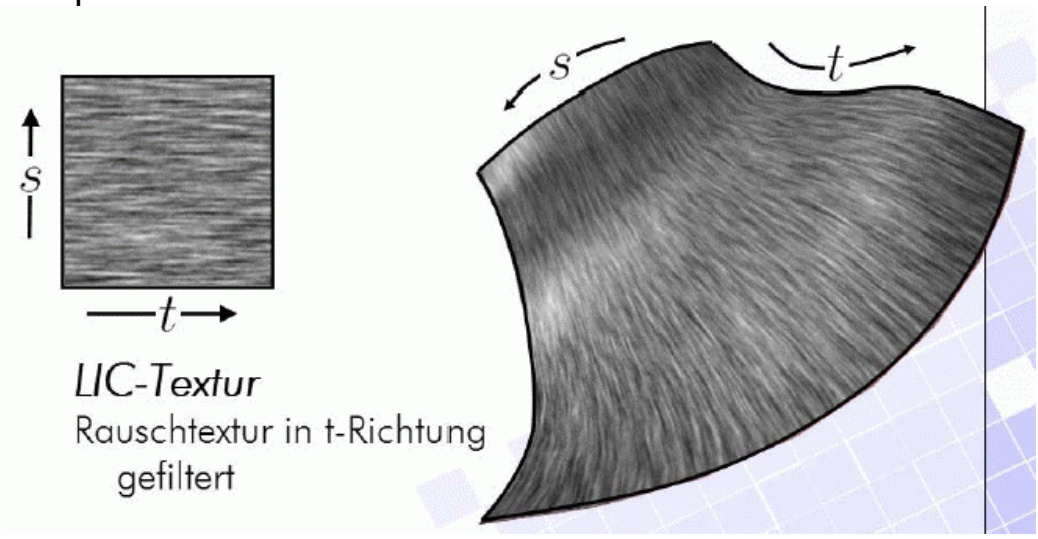
- *Beispieltexturen*
  Farbtextur ersetzt den ambienten Term im Phong-Modell

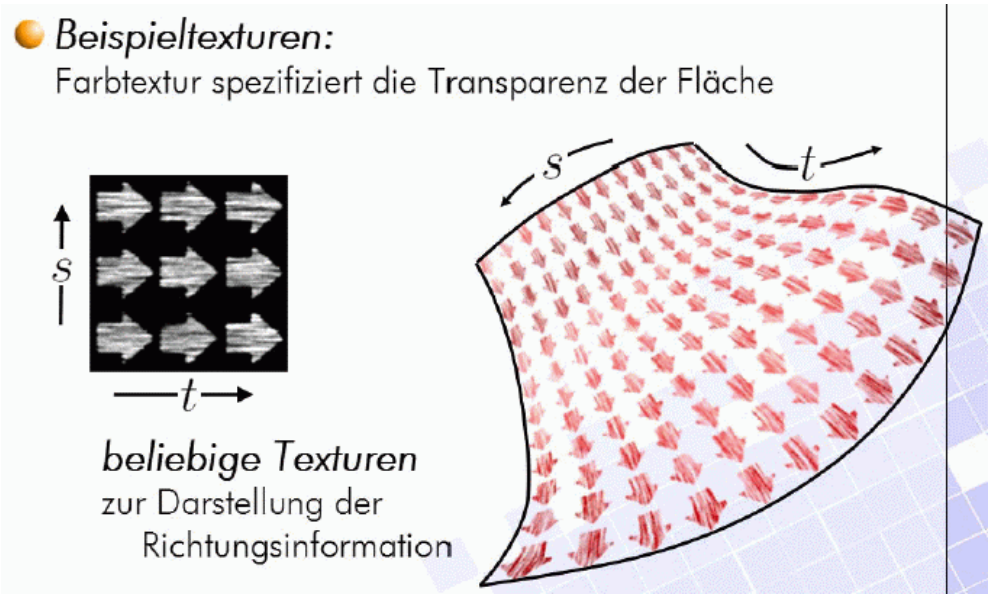


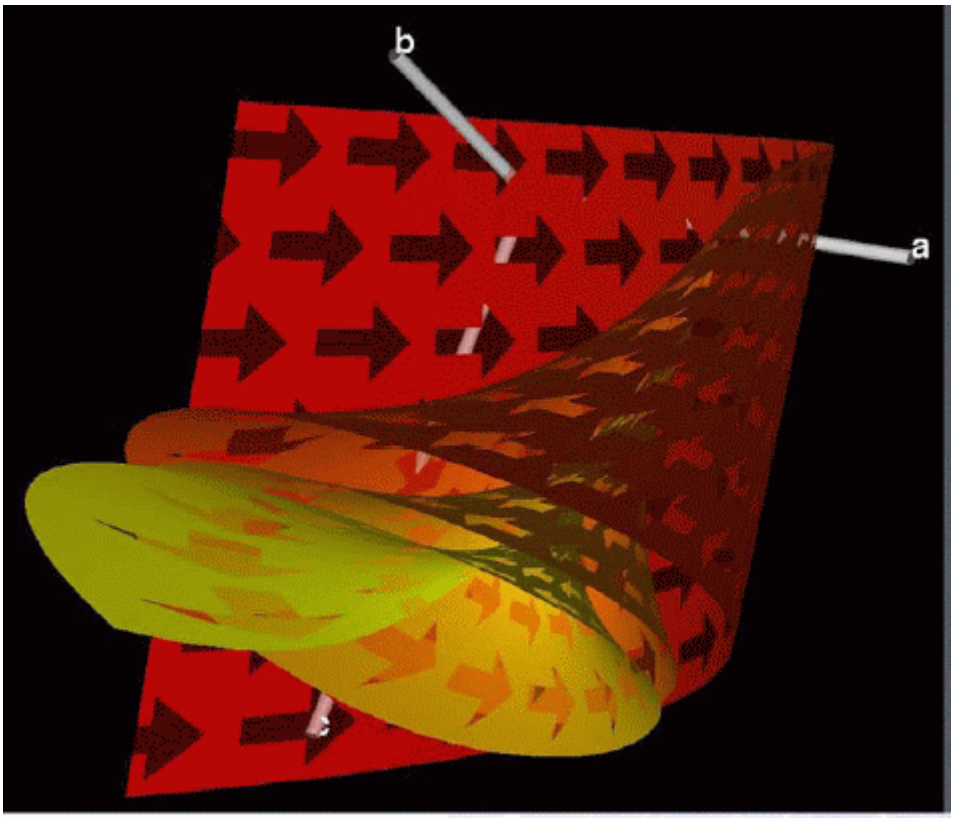
Vektorpfeile



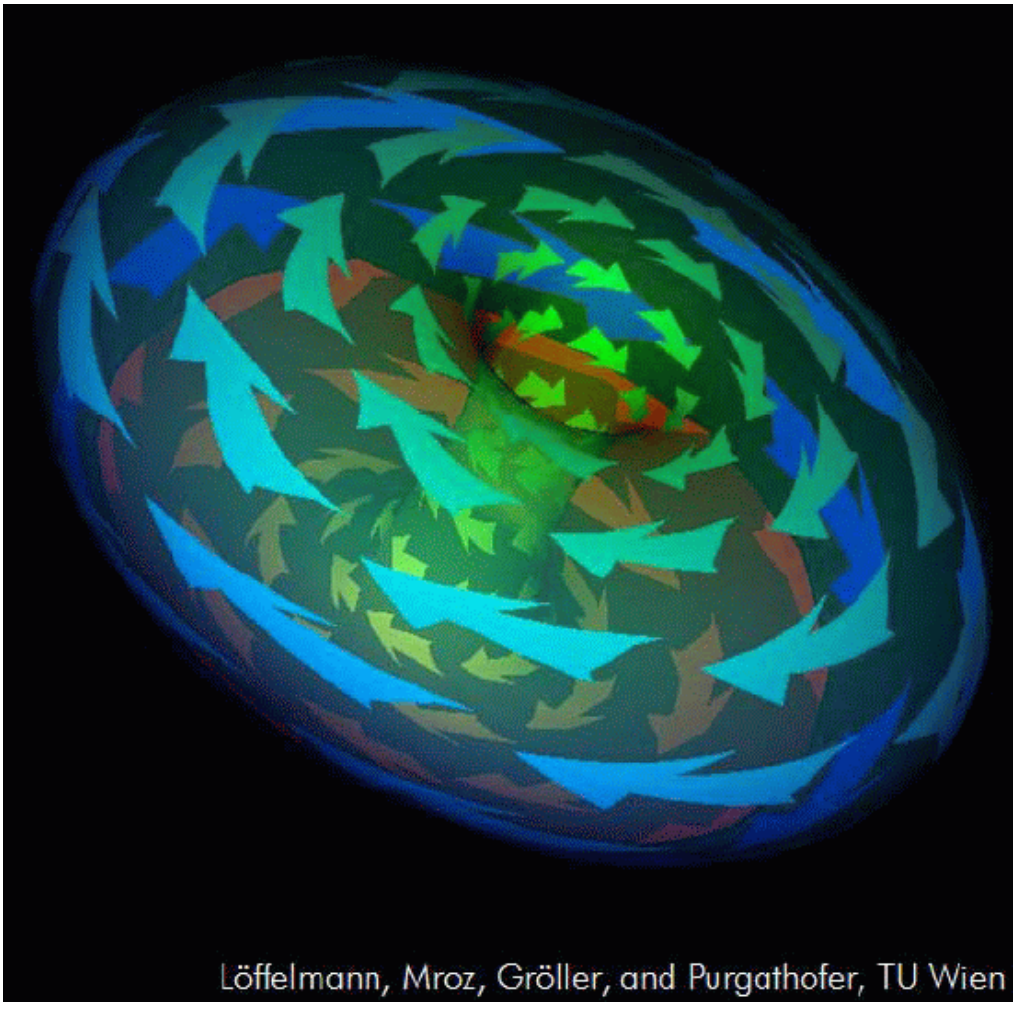
Beispieltexturen







Löffelmann, Mroz, Gröller, and Purgathofer, TU Wien



Löffelmann, Mroz, Gröller, and Purgathofer, TU Wien

Time Surfaces

- *Definition: Time Surfaces (Flächen gleicher Zeit)*

  Time Surfaces zeigen die durch die Strömung verursachte Deformation einer Startfläche zu einem bestimmten Zeitpunkt.

Startfläche $\mathbf{F}_0(u,v)$

Time Surface $\mathbf{F}_t(u,v)$

$$\frac{\partial \mathbf{F}_t(u,v)}{\partial t} = \vec{v}(\mathbf{F}_t(u,v))$$

Jeder Punkt der Fläche bewegt sich auf einer Partikelbahn:

$\vec{f}(t) = \mathbf{F}_t(U,V)$ *mit* $U, V = \text{const.}$

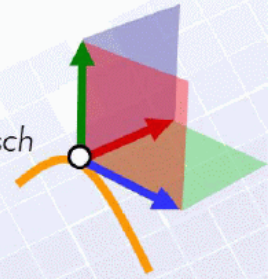
Wahl der Startfläche:

Die Startfläche sollte möglichst orthogonal zur Strömung sein:

- *Manuelle Wahl* (als *Flow Field Probe*)
- *Automatrische Wahl* (ausgehend von einem Punkt):

  Frenet Frame: Ideale Startfläche wird durch die Normale und die Binormale aufgespannt.
  - Bestimme zunächst eine Integralkurve in Richtung der Binormalen.
  - Bestimme dann eine „Strömungsfläche“ in Richtung der Normalen
- *Übliche Alternative:* *Semiautomatisch*

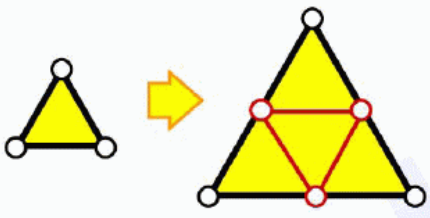
  Eine *Ebene* wird automatisch entlang der Normalen und Binormalen ausgerichtet



- *Problematik: Divergenz* $\operatorname{div} \vec{v} > 0$

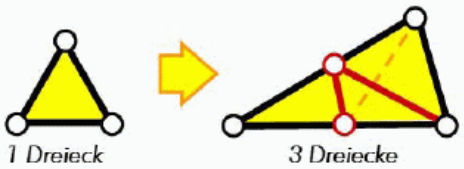
  analog zu Streamsurfaces: Kanten werden zu lang.

  Dreiecksfläche:
  - *Fall 1*: 3 Kanten sind zu lang



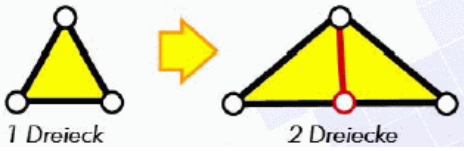
1 Dreieck *4 Dreieck*

- *Fall 2*: 2 Kanten sind zu lang:



  - Wähle von den beiden möglichen Diagonalen des Vierecks die kürzere

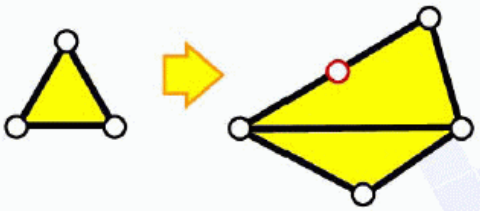
- *Fall 3*: Nur 1 Kante ist zu lang:



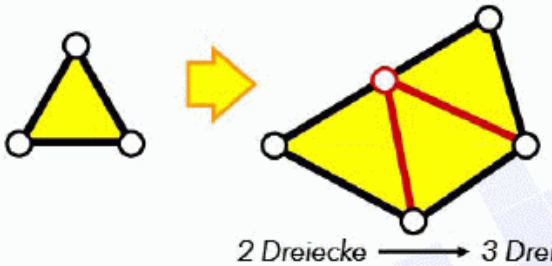
Dreiecksfläche: Optimierung

- *Fall 2*: 2 Kanten sind zu lang:

  Falls eines der Nachbardreiecke vom Typ 2 oder 3 ist:



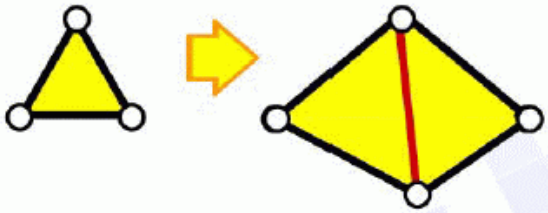
  - *Untersuche die Nachbardreiecke der langen Kanten*



Mit dieser Methode wächst die Anzahl der Dreiecke nicht so schnell.

- *Fall 3*: Nur 1 Kante ist zu lang:

  Falls eines der Nachbardreiecke vom Typ 2 oder 3 ist:



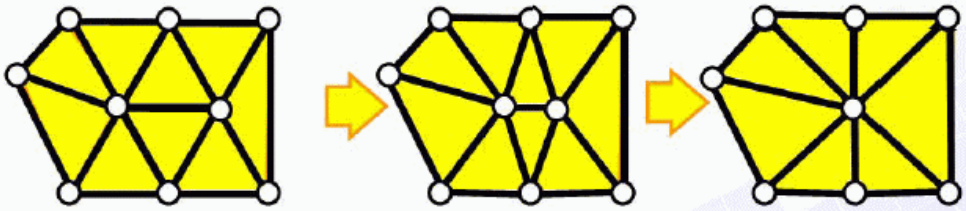
  - *Untersuche das Nachbardreieck der langen Kante*

Hier: Anzahl der Dreiecke bleibt konstant.

- *Problematik: Konvergenz* $\operatorname{div} \vec{v} < 0$

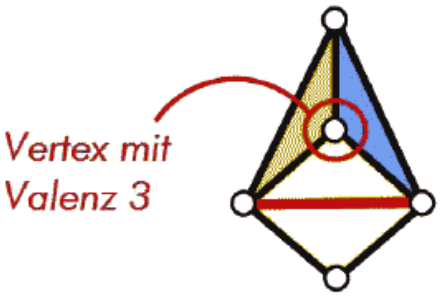
  analog zu Streamsurfaces: Kanten werden zu kurz.

Edge Collapse für Dreiecksnetze



- *Achtung:* nicht alle kurzen Kanten können kollabiert werden!

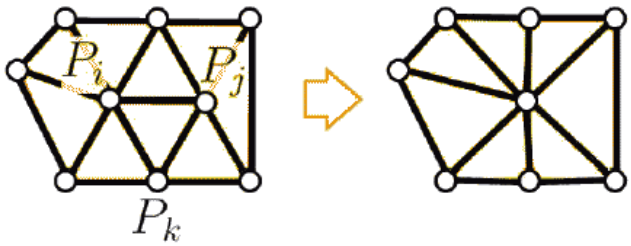
- *Beispiel für einen ungültigen Edge Collapse*



- Die rote Kante darf nicht kollabiert werden, ansonsten entsteht eine Überfaltung des blauen und des grünen Dreiecks!
- *Valenz* = Anzahl der Kanten, die von einem Vertex ausgehen.

- *Vor dem Edge Collapse muß geprüft werden, ob die Topologie des Dreiecksnetzes erhalten bleibt!*

Edge Collapse

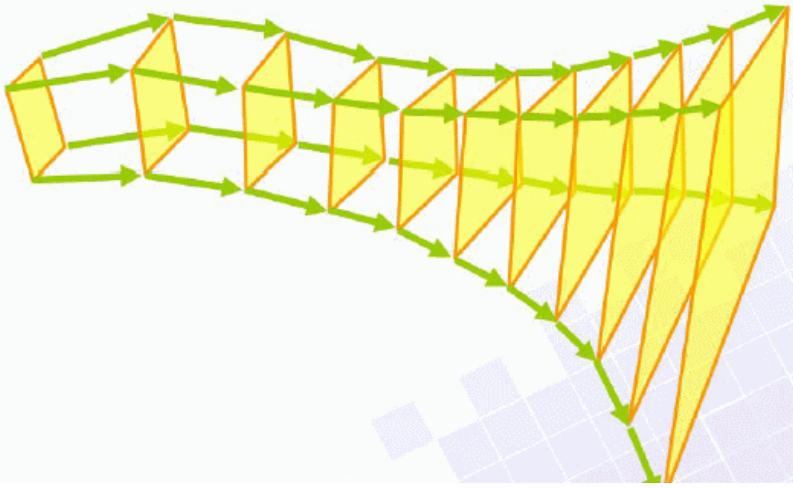


- Bevor ein *Edge-Collapse* $P_i\,P_j$ ausgeführt wird, müssen folgende Bedingungen geprüft werden:
  1. Für jeden Punkt P_k, der sowohl mit P_i als auch mit P_j benachbart ist, muß das Dreieck ($P_k\,P_i\,P_j$) existieren.
  2. Wenn P_i und P_j beides Randpunkte sind, dann ist die Kante $P_i\,P_j$ eine Randkante

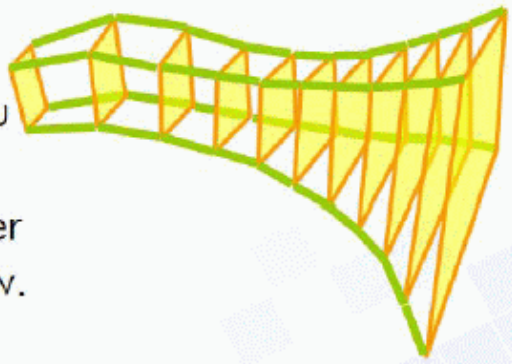
3. Wenn P_i und P_j keine Randpunkte sind, muß das Dreiecksnetz mehr als 4 Punkte haben.
Wenn einer der beiden Punkte P_i oder P_j ein Randpunkt ist, muß das Dreiecksnetz aus mehr als 3 Punkten bestehen.

Volume Flow

- *Idee:*
  Kombiniere *Stream Surfaces* mit *Time Surfaces:*
  Betrachte das Volumen.

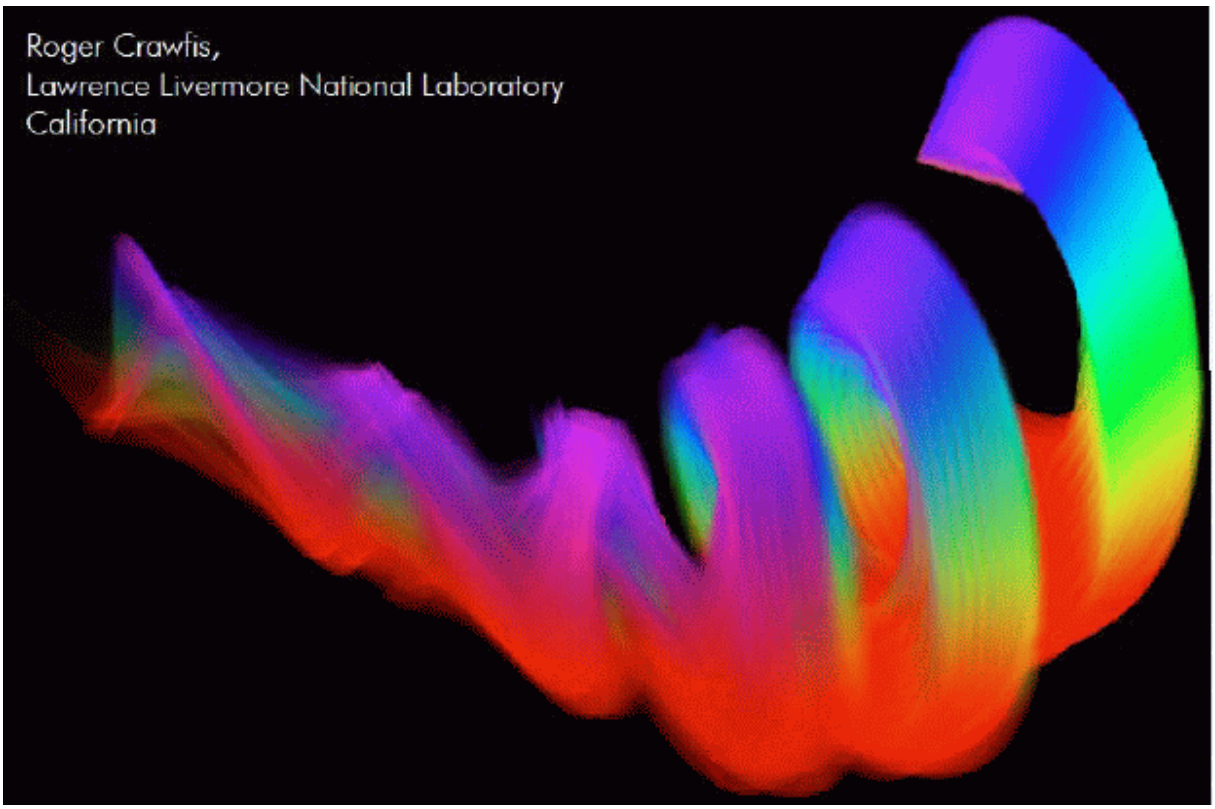


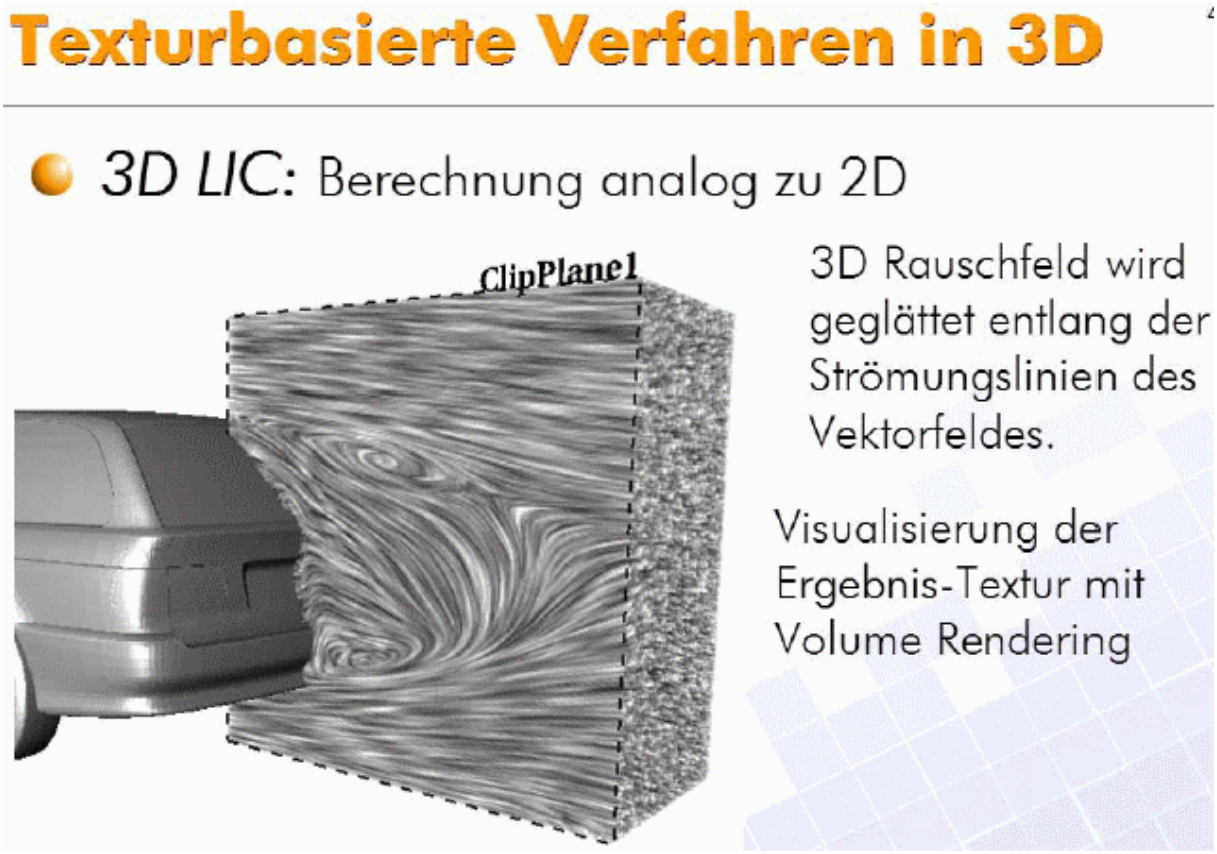
- Startpolygon (Rechteck) wird durch die Strömung bewegt.
- *Splitting und Rippling:* analog zu den Streamsurfaces wird das Volumen in Bereichen mit großer positiver Divergenz unterteilt bzw. zerrissen
- Kein *Merging*



- Innerhalb des Flow Volumes wird ein (konstanter) Skalarwert angenommen (Farbe und Transparenz)
- Das *Flow Volume* wird in Tetraeder unterteilt und mit *Volume Rendering* (*Shirley & Tuchman*) dargestellt.

Volume Flow – Ergebnis (Beispiel):





(Rezk-Salama, o.J.)