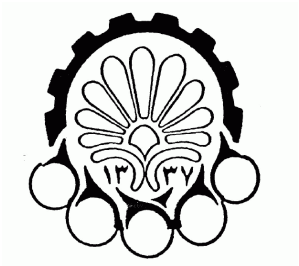


بسمه تعالی

پروژه پایانی درس سیستم های چند رسانه ای

عنوان پروژه:

پیاده سازی متد Horn-Schunk برای Optical Flow

ارائه دهنده : سپند حقیقی

شماره دانشجویی : ۹۰۲۳۰۷۶

زمستان ۹۳

در این پروژه از الگوریتم **Horn-Schunck** برای پیاده سازی **Optical Flow** استفاده شده است که در آن هدف استخراج معادله ای برای مرتبط کردن تغییرات روشنایی یک تصویر در نقطه **(x,y)** با الگوی حرکتی روشنایی تصویر است **E(x,y,t)** نشان دهنده روشنایی نقطه ی **(x,y)** در زمان **t** است حال فرض می کنیم که زمانی که یک الگو حرکت می کند روشنایی یک نقطه خاص از این الگو تغییر نمی کند :

$$\frac{dE}{dt} = 0$$

$$\frac{\partial E}{\partial x}\frac{dx}{dt} + \frac{\partial E}{\partial y}\frac{dy}{dt} + \frac{\partial E}{\partial t} = 0$$

معرفی سرعت های عمودی و افقی و معادله خطی تغییرات جزیی جریان نوری :

$$u = \frac{dx}{dt}, v = \frac{dy}{dt}$$

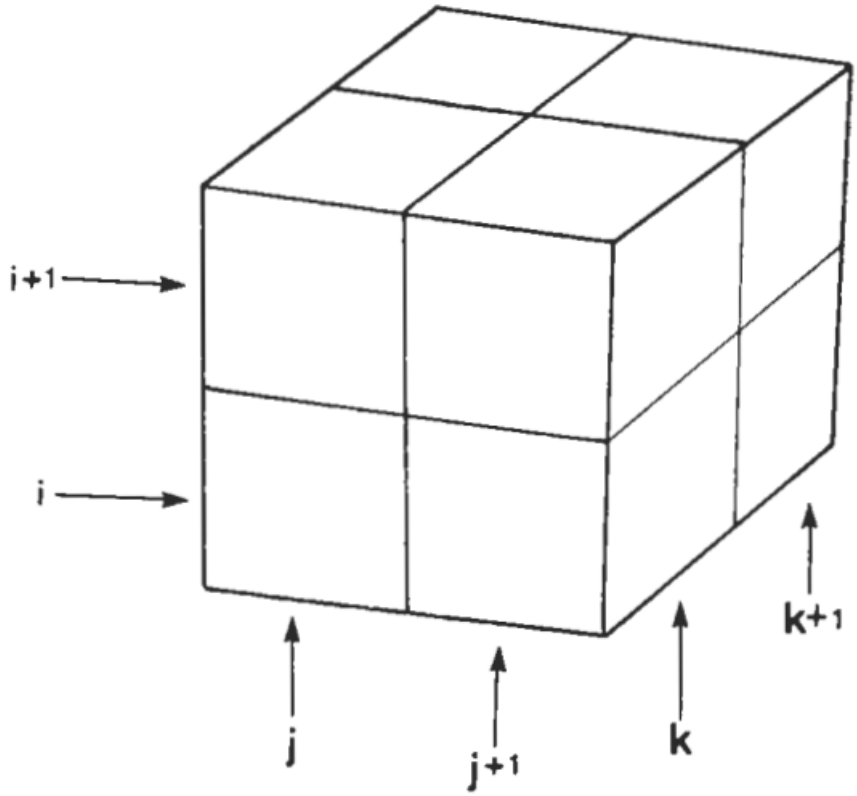
$$E_x u + E_y v + E_t = 0$$

در ادامه این الگوریتم با استفاده از معادلات زیر مشتقات جزیی **E(x,y,t)** را تخمین می زند : (این تخمین به صورت میانگین چهار تفاضل متوالی در یک همسایگی ۸تایی است (شکل-۱)

$$E_x \approx \frac{1}{4}(E_{i,j+1,k} - E_{i,j,k} + E_{i+1,j+1,k} - E_{i+1,j,k} + E_{i,j+1,k+1} - E_{i,j,k+1} + E_{i+1,j+1,k+1} - E_{i+1,j,k+1})$$

$$E_y \approx \frac{1}{4}(E_{i+1,j,k} - E_{i,j,k} + E_{i+1,j+1,k} - E_{i,j+1,k} + E_{i+1,j,k+1} - E_{i,j,k+1} + E_{i+1,j+1,k+1} - E_{i,j+1,k+1})$$

$$E_t \approx \frac{1}{4}(E_{i,j,k+1} - E_{i,j,k} + E_{i+1,j,k+1} - E_{i+1,j,k} + E_{i,j+1,k+1} - E_{i,j+1,k} + E_{i+1,j+1,k+1} - E_{i+1,j+1,k})$$



شکل-۱

در مرحله بعد نیاز به تخمین لاپلاسین سرعت جریان نوری داریم که تخمین زیر قابل قبول است :

$$\nabla^2 u \simeq \kappa(\bar{u}_{i,j,k} - u_{i,j,k})$$

$$\nabla^2 v \simeq \kappa(\bar{v}_{i,j,k} - v_{i,j,k})$$

$$\bar{u}_{i,j,k} = \frac{1}{6}(u_{i-1,j,k} + u_{i,j+1,k} + u_{i+1,j,k} + u_{i,j-1,k}) + \frac{1}{12}(u_{i-1,j-1,k} + u_{i-1,j+1,k} + u_{i+1,j+1,k} + u_{i+1,j-1,k})$$

$$\bar{v}_{i,j,k} = \frac{1}{6}(v_{i-1,j,k} + v_{i,j+1,k} + v_{i+1,j,k} + v_{i,j-1,k}) + \frac{1}{12}(v_{i-1,j-1,k} + v_{i-1,j+1,k} + v_{i+1,j+1,k} + v_{i+1,j-1,k})$$

و در مرحله بعد با استفاده از روش تکراری گوس-سایدل سرعت ها را در راستای کمتر شدن MSE آپدیت می کنیم :

که در آنها ξ_b نرخ تغییر روشنایی تصویر و ξ_c خطای اندازه گیری در تغییر حالت از حالت سکون در جریان نوری است.

$$\xi_b = E_x u + E_y v + E_t$$

$$\xi_c^2 = (\frac{\partial u}{\partial x})^2 + (\frac{\partial u}{\partial y})^2 + (\frac{\partial v}{\partial x})^2 + (\frac{\partial v}{\partial y})^2$$

$$\xi^2 = \iint \left(\alpha^2 \xi_c^2 + \xi_b^2 \right) dx dy$$

$$u^{n+1} = \frac{\bar{u}^{n} - E_x [E_x \bar{u}^{n} + E_y \bar{v}^{n} + E_t]}{\alpha^2 + {E_x}^2 + {E_y}^2}$$

$$v^{n+1} = \frac{\bar{v}^{n} - E_y [E_x \bar{u}^{n} + E_y \bar{v}^{n} + E_t]}{\alpha^2 + {E_y}^2 + {E_x}^2}$$

توضیح توابع در متلب :

۱- start.m : اسکریپت شروع برنامه ، با اجرای این اسکریپت از کاربر دو حالت ورودی فیلم از وبکم یا فایل پرسیده شده سپس در هر حالت توابع مربوط به آن ها را فراخوانی می کند.

۲-webcam.m : در این تابع ابتدا مدل وبکم کاربر و فرمت های قابل پشتیبانی با آن نمایش داده می شود بعد از انتخاب کیفیت توسط کاربر ، وبکم فراخوانی شده و در هر مرحله فریمی از آن گرفته می شود و پردازش برروی آن فریم صورت می گیرد و در انتها تصویر وبکم به همراه میدان های حرکتی و ماکسیمم تغییرات سرعتی آن نمایش داده می شود.

۳-readframe.m : این تابع در شی ای از نوع فایل تصویری و شماره فریم به عنوان ورودی گرفته شده و تصویر آن فریم در خروجی قرار می گیرد.

٤-fromvideo.m : این تابع فایل تصویری و کد نوع تصویر (رنگی-خاکستری) را به عنوان ورودی گرفته و پس از استخراج فریم های آن به وسیله ی تابع readframe.m و پردازش های لازم برروی آن مانند تابع webcam.m در انتها تصویر فیلم به همراه میدان های حرکتی و ماکسیمم تغییرات سرعتی آن را نمایش می دهد.

۵-**opticalflow.m**: *این تابع بردار تصاویر ، آلفا(وزن خطا) و تعداد تکرار های لازم برای گوس-سایدل را به عنوان ورودی گرفته و محاسبات روش* Horn-Schunk *را برروی فریم های متوالی انجام داده و سرعت های افقی و عمودی را در خروجی قرار می دهد.*