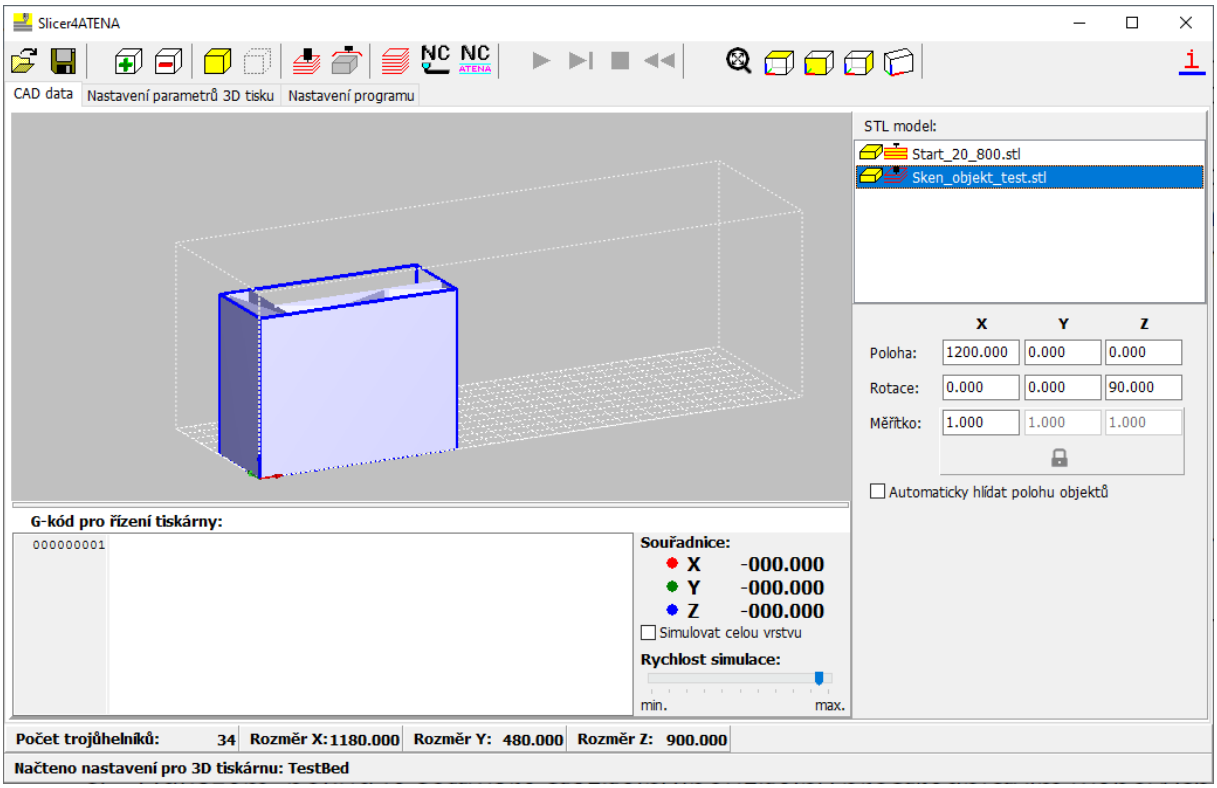
Technická dokumentace programu Slicer4ATENA

Program Slicer4ATENA je určení ke generování tzv. G-kódu prořízení drah tiskové hlavy 3D tiskárny, primárně pro tisk z cementových směsí a podobných materiálů. Strategie řízení drah tiskové hlavy vychází z CAD modelu tištěného objektu. Předpokládá se, že se bude nejčastěji tisknout betonová stěna pouze na šířku jednoho průměru trysky, tj. jednoho perimetru. V rámci řešitelského týmu bylo dohodnuto, že 3D objekty k tisku budou převedeny na plošné modely, kde vymodelovaná plocha tvoří osovou plochu budoucí zdi objektu. 3D model objektu, který se bude tisknout je tedy tvořen pouze plochami nulové tloušťky, které dohromady neuzavírají výsledný objem.

To ale přináší problém s přípravou G-kódu pro řízení pohybů tiskové hlavy. Nelze použít existující slicery, jako např. Slic3r, PrusaSlicer, CuraEngine apod. Tyto programy jsou primárně připraveny pro 3D modely tvořící objemová tělesa, příp. plně uzavřená plošná tělesa a příprava tiskových dat z ploch nulové tloušťky je přinejmenším problematická, v řadě případů nemožná. Tyto programy také neumožňují plně optimalizovat a přizpůsobovat strategie tisku požadavkům pro 3D tisk z cementových směsí, kdy je dále třeba respektovat další omezení způsobu řízení tiskové hlavy. Sem patří zejména optimalizace trajektorie tiskové hlavy a minimalizace přejezdů, protože vzhledem k povaze tiskového materiálu může docházet k jeho samovolnému úniku a znečišťování prostředí stavby.

V rámci přípravy jedné tiskové úlohy je možné načíst i více objektů ve formátu STL. Načtení STL dat je zajišťováno pomocí procedur *NactiSoubor(const JmenoSouboru: string)* nebo *NactiProjekt(const JmenoSouboru: string)*. Program akceptuje STL objekty jak v binární, tak ASCII reprezentaci. Načtené objekty se zobrazují v grafické oblasti programu i v seznamu na pravé straně téhož okna, viz obr. 1.

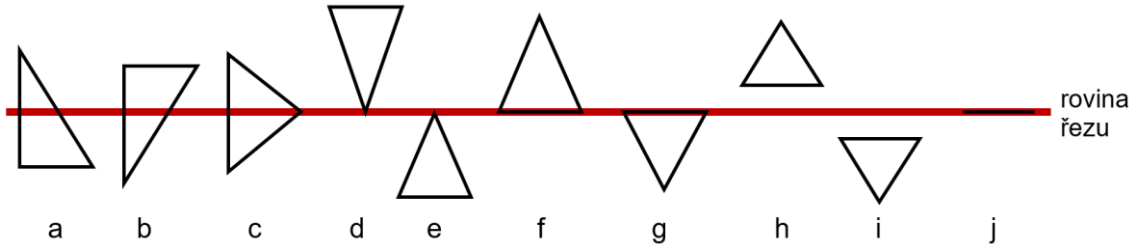


Obr. 1 Program Slicer4ATENA – jeho hlavní pracovní prostředí

V rámci přípravy úlohy je možné definovat pro každý prvek transformaci v pracovním prostoru tiskárny (posun, rotaci a měřítko) a dále zda bude vybraný prvek zpracováván pro tisk, či zda se jedná např. o dříve vytištěný objekt, kterému se musí tisková hlava vyhnout. Tuto informaci spolu s informací o viditelnosti objektu znázorňují ikony v seznamu prvků vlevo od jejich názvů.

V okamžiku, kdy jsou všechny prvky pro následný 3D tisk na svém místě, je možné přistoupit k výpočtu jednotlivých vrstev tak, jak se budou objekty postupně stavět – funkce *VypocitejRez(const Vrstva: Integer; var RezZ: TDataRezu; const Start: Boolean): Boolean*.

Prvním krokem je rozřezání všech STL objektů rovinami rovnoběžnými ze základnou XY s roztečí definovanou jako „Tloušťka vrstvy“ v záložce „Nastavení tiskových parametrů 3D tisku“. Protože je objekt ve formátu STL tvořen pouze trojúhelníkovými plochami, může v každém řezu pro každý trojúhelník objektu nastat jen jeden z případů zobrazených na obr. 2.



Obr. 2 Možné situace při generování řezu STL daty

V případě, že rovina řezu protíná daný trojúhelník STL plochy (obr. 2a, b, c), jsou vypočteny průsečíky trojúhelníku s rovinou řezu a vzniklá úsečka je uložena do paměti pro další zpracování. Pokud má trojúhelník s rovinou řezu společný pouze jeden bod (obr. 2d, e), není třeba tyto trojúhelníky (resp. dané body) pro daný řez zahrnout pro další zpracování v tomto řezu, podobně jako trojúhelníky ležící mimo rovinu řezu (obr. 2h, i), nebo trojúhelník ležící celý v rovině řezu (obr. 2j). Zbývají tedy trojúhelníky, které mají s rovinou řezu společnou jednu stranu (obr. 2f, g). Tato strana by měla být také uložena pro další zpracování, problematické ale je, že by v datech mohly vznikat duplicitní úsečky díky navazujícím trojúhelníkům. Proto algoritmus použitý v programu Slicer4ATENA používá pro další zpracování pouze stranu toho trojúhelníku, jehož třetí vrchol leží pod rovinou řezu (obr. 2g).

Výsledkem této první fáze generování řezů je tedy sada jednotlivých úseček uložených v datové struktuře *TDataRezu*, což je dynamické datové pole. Druhým krokem je jejich optimalizace – uspořádání a pospojování navazujících úseček do jednoho či více polygonů, dle složitosti objektu. Tyto polygony by již mohly být použity pro generování tzv. G-kódu pro řízení pohybů tiskové hlavy, z důvodů uvedených výše je ale v případě tisku z betonu nutná další optimalizace. Základ této optimalizace spočívá v určení délek přejezdů v jedné vrstvě (pokud je v ní více polygonů), stanovení pořadí tisku polygonů a v případě uzavřených polygonů i stanovení místa napojení tak, aby délka přejezdů byla minimální. Dále se zohledňují a minimalizují i přejezdy mezi vrstvami. Další fází optimalizace je zjednodušení polygonů, kdy i rovinná stěna může být v STL datech reprezentována množstvím trojúhelníků. Řez takovou stěnou pak dá polygon úseček, které ale mají všechny stejnou směrnici, tzn., že takový polygon pak může být nahrazen jen jednou úsečkou. Optimalizace probíhá též na základě nastavení uživatelských parametrů, které je možné nastavovat / měnit v záložce programu „Nastavení parametrů 3D tisku“. Celé optimalizace polygonu je počítána pomocí procedur *OptimalizujData(var RezZ: TDataRezu; const ZacX, ZacY: Double)* a *SestavPolygony(var RezZ: TDataRezu; const ZacX, ZacY: Double)*.

Dále je možné volitelně v parametrech tisku zapnout i optimalizaci, která prokládá polygonem úseček kruhové oblouky, pokud je chyba proložení menší, než zadaná. Z principu formátu dat STL byly totiž původně matematické válcové plochy z CAD modelu nahrazeny polygonálními plochami složenými z mnoha trojúhelníků. Kvalita, tj. jemnost nebo hrubost, těchto ploch je dána nastavením parametrů exportu do STL formátu v CAD programech. Toto prokládání kruhových oblouků je však řešeno mimo dříve zmíněné optimalizační procedury, protože výstup dat pro simulační software ATENA kruhové interpolace nepodporuje a v tomto případě jsou výstupem vždy pouze optimalizované polygony úseček.

Posledním krokem je samotné generování G-kódu. V principu se jedná o převedení optimalizovaných polygonů do „řeči“ stroje pro 3D tisk. V záložce „Nastavení tiskových parametrů 3D tisku“ je možné upravit rychlosti tisku, způsob řízení extrudéru tiskárny a řadu dalších parametrů ovlivňujících vlastní tisk. Ověřenou konfiguraci parametrů je možné uložit pro jednodušší nastavení dalších tisků. Takže po uspořádání všech tištěných objektů ve virtuálním pracovním prostoru stroje v programu Slicer4ATENA je příprava G-kódu poté záležitostí jen dvou kliknutí myší. Generování G-kódu zajišťují procedury *TB_NCkodClick(Sender: TObject)*, příp. *TB_NC_AtenaClick(Sender: TObject)*.
Vygenerovaný G-kód je možné ještě před vlastním nahráním a spuštěním ve stroji v programu Slicer4ATENA simulovat. Klíčová procedura pro simulaci G-kódu v 3D zobrazení programu je procedura *ProvedRadek(const Radek: string)*.

Funkčnost programu byla ověřena jednak použitím při generování G-kódů pro vlastní 3d tisk z cementových směsí, jednak načtením G-kódu upraveného pro SW ATENA do tohoto prostředí pro nelineární analýzu betonových a železobetonových konstrukcí.

Celý program je napsaný v jazyce Object Pascal a zkompilován ve vývojovém prostředí Delphi 11 jako 32 bitová aplikace pro OS Windows, využívající prvky Windows VCL. Při vývoji programu byla použita open-source komponenta GLScene, která slouží pro zobrazování a manipulaci s 3D objekty na obrazovce – pracovní prostor tiskárny se souřadným systémem, STL data a data řezů a 3D simulaci. Pro zobrazování G-kódu a zvýrazňování aktuálně simulovaného řádku programu je dále použita komponenta open-source SynEdit. Poslední použitou komponentou mimo vlastní balík vývojového prostředí Delphi 11 je komponenta RealEdit, což je v podstatě nadstavba klasického editačního prvku Edit, která omezuje možnost zadávání uživatelských vstupů na jen předem definované – např. pouze reálná čísla, jen celá kladná čísla apod. Vlastní zdrojový kód programu Slicer4ATENA je v příloze.

Vývoj tohoto programu byl podpořen projektem TAČR TREND „Simulace a navrhování konstrukcí z digitálního betonu“ (digiBeton), kód projektu FW06010422.